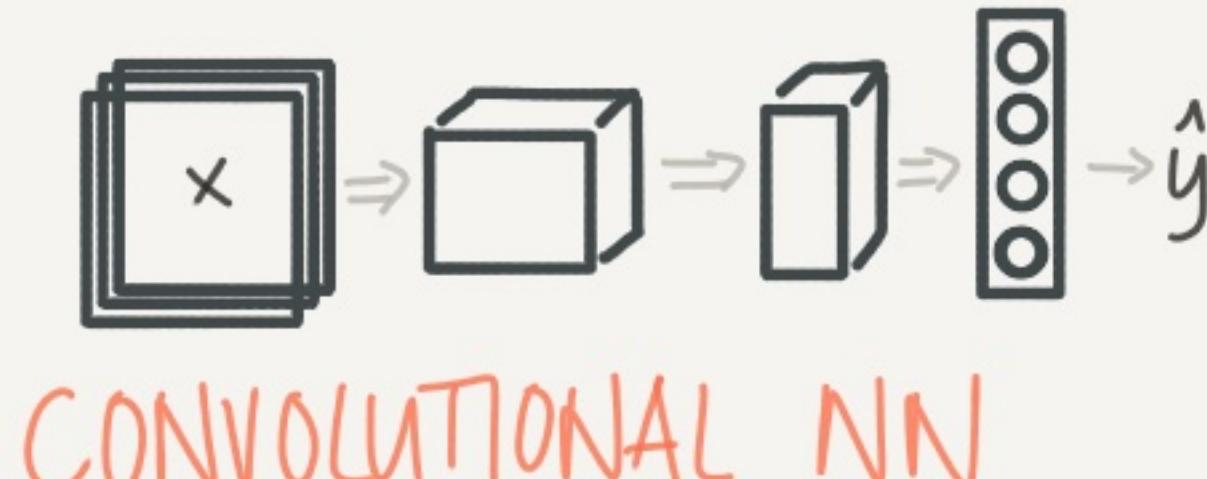
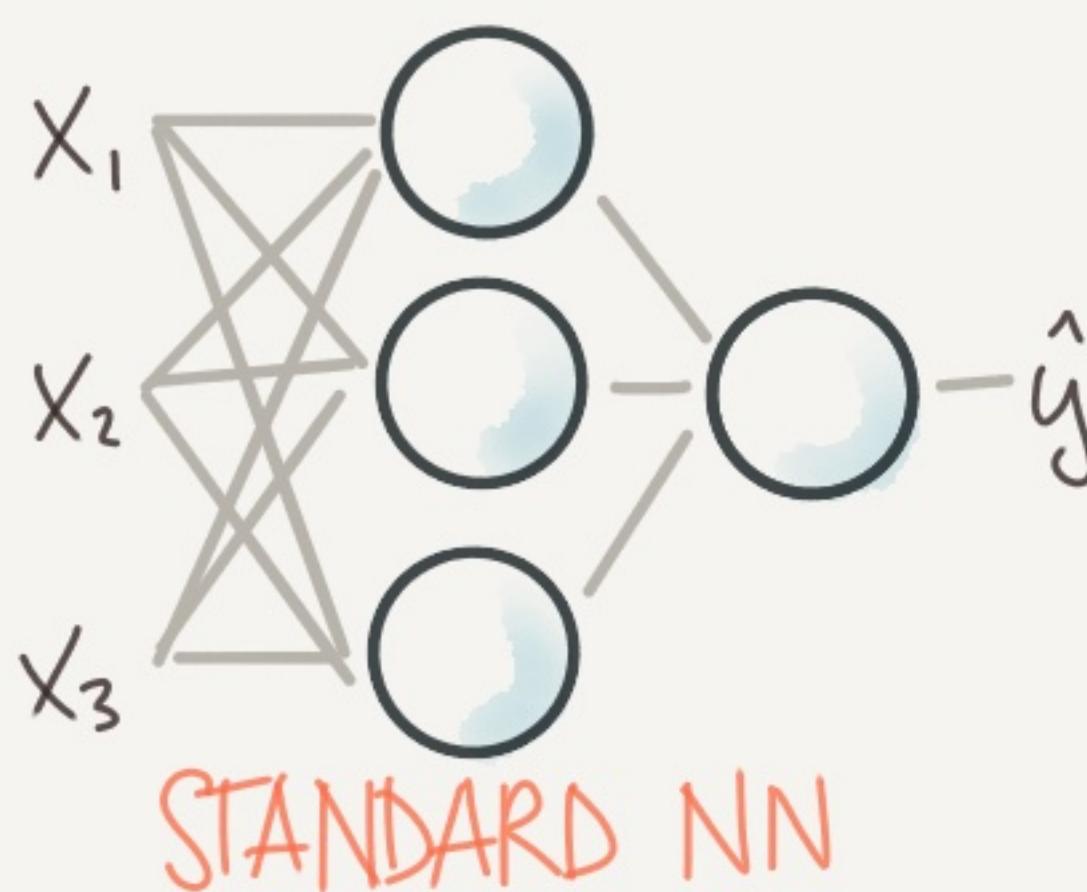


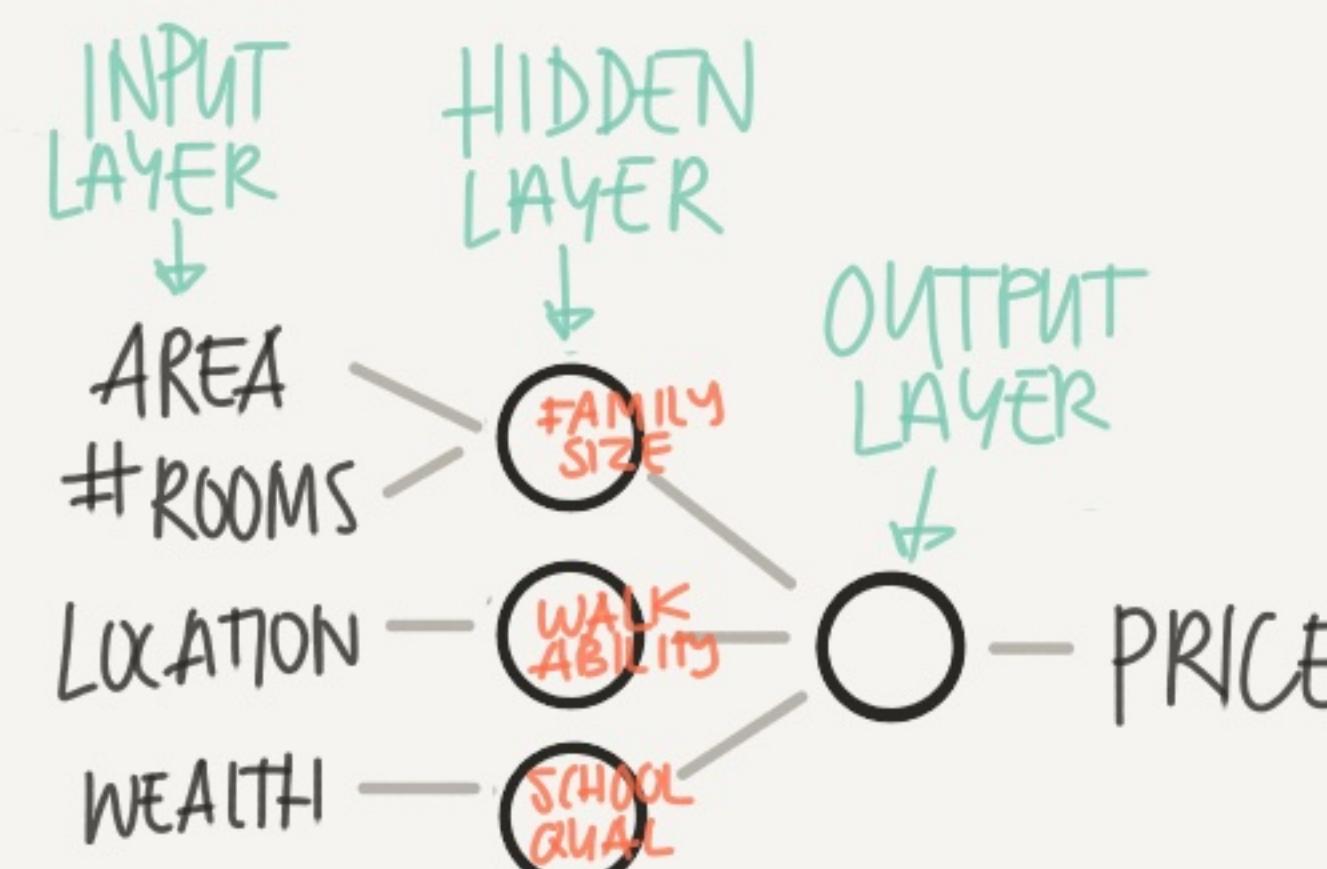
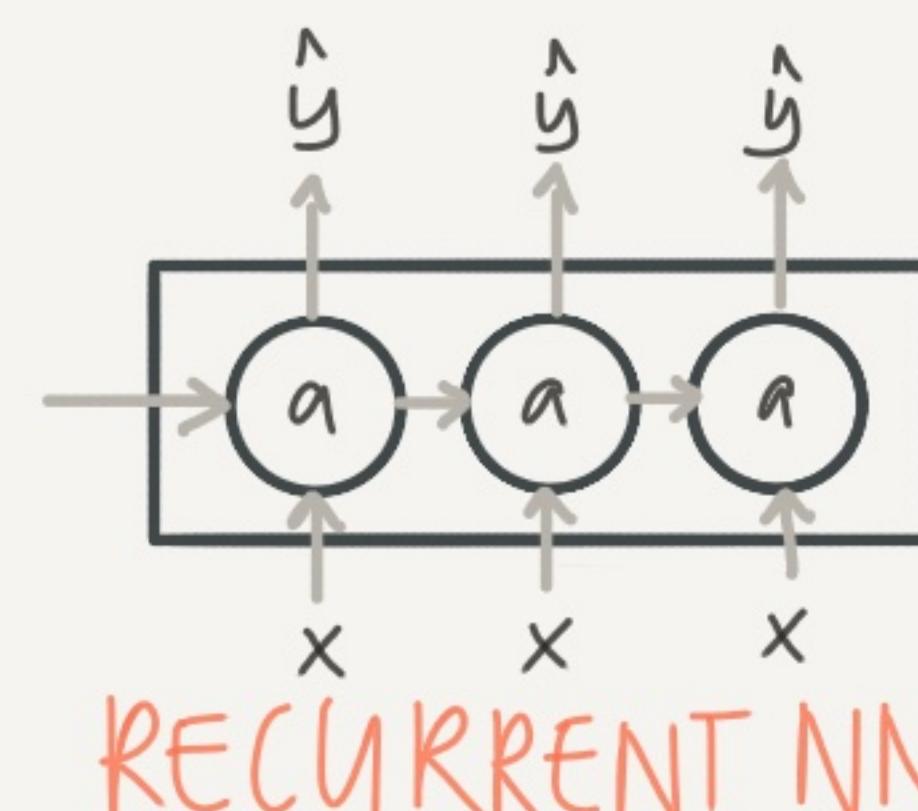
# INTRO TO DEEP LEARNING

## SUPERVISED LEARNING

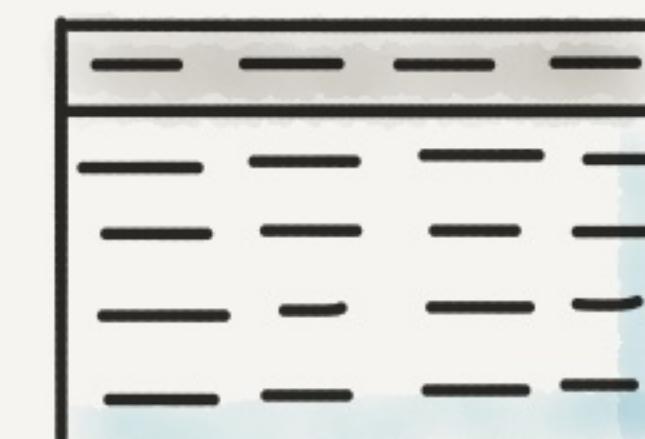
INPUT: $X$	OUTPUT: $y$	NN TYPE
HOME FEATURES	PRICE	STANDARD NN
AD+USER INFO	WILL CLICK ON AD (0/1)	
IMAGE	OBJECT (1...1000)	CONV. NN (CNN)
AUDIO	TEXT TRANSCRIPT	RECURRENT NN (RNN)
ENGLISH	CHINESE	
IMAGE/RADAR	POS OF OTHER CARS	CUSTOM/HYBRID



## NETWORK ARCHITECTURES



NNs CAN DEAL WITH BOTH STRUCTURED & UNSTRUCTURED DATA



STRUCTURED

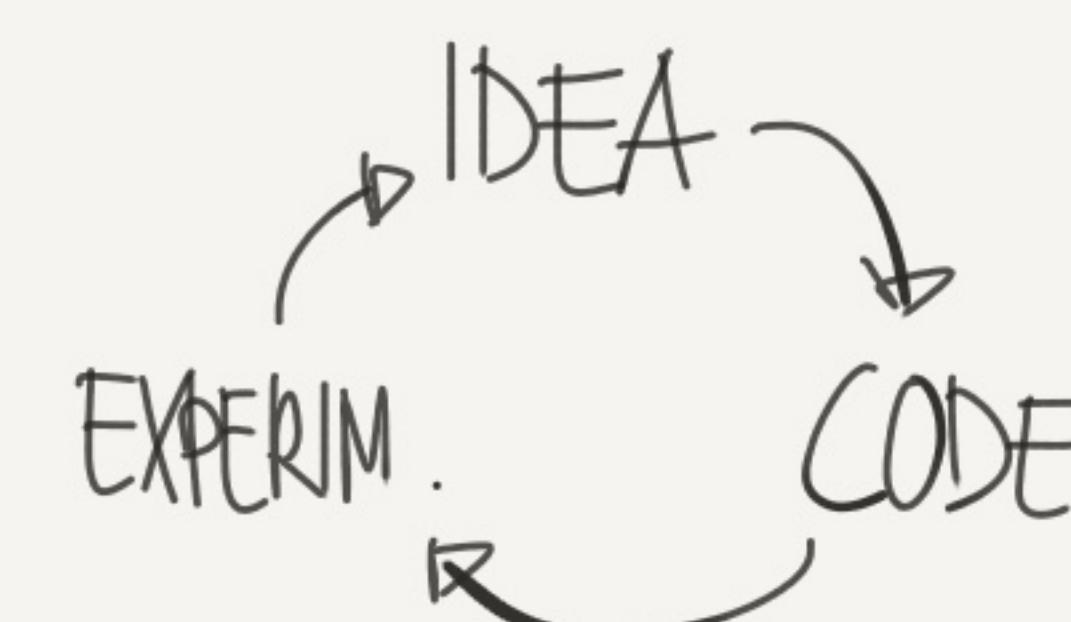
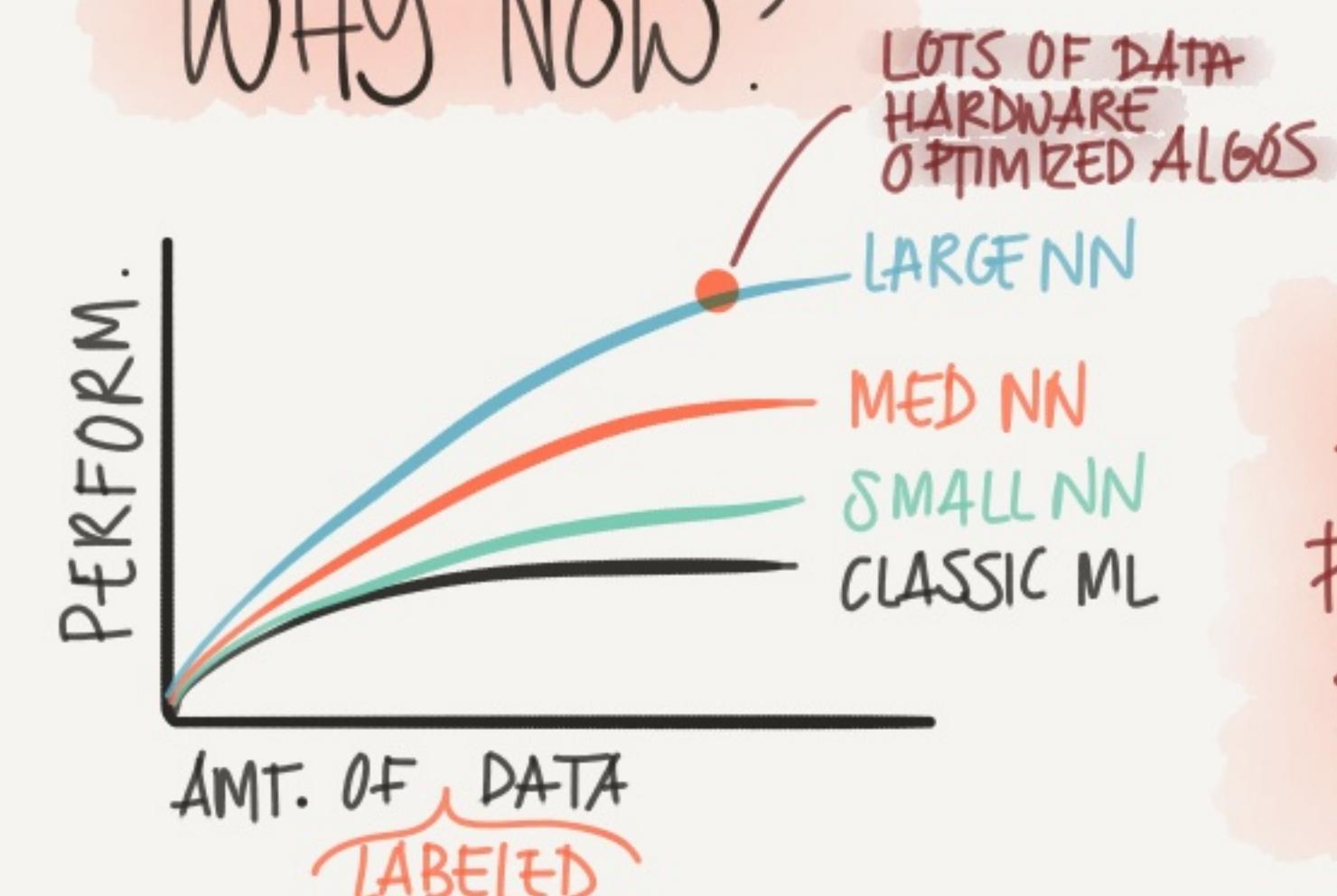


"THE QUICK BROWN FOX"

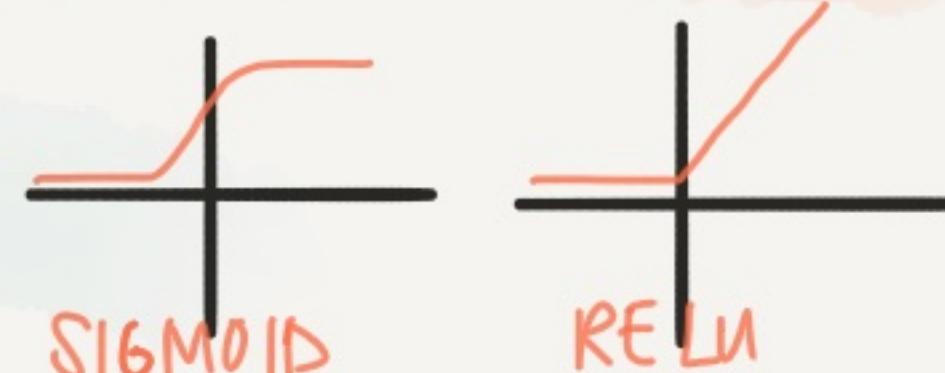
UNSTRUCTURED

HUMANS ARE GOOD  
AT THIS

WHY NOW?



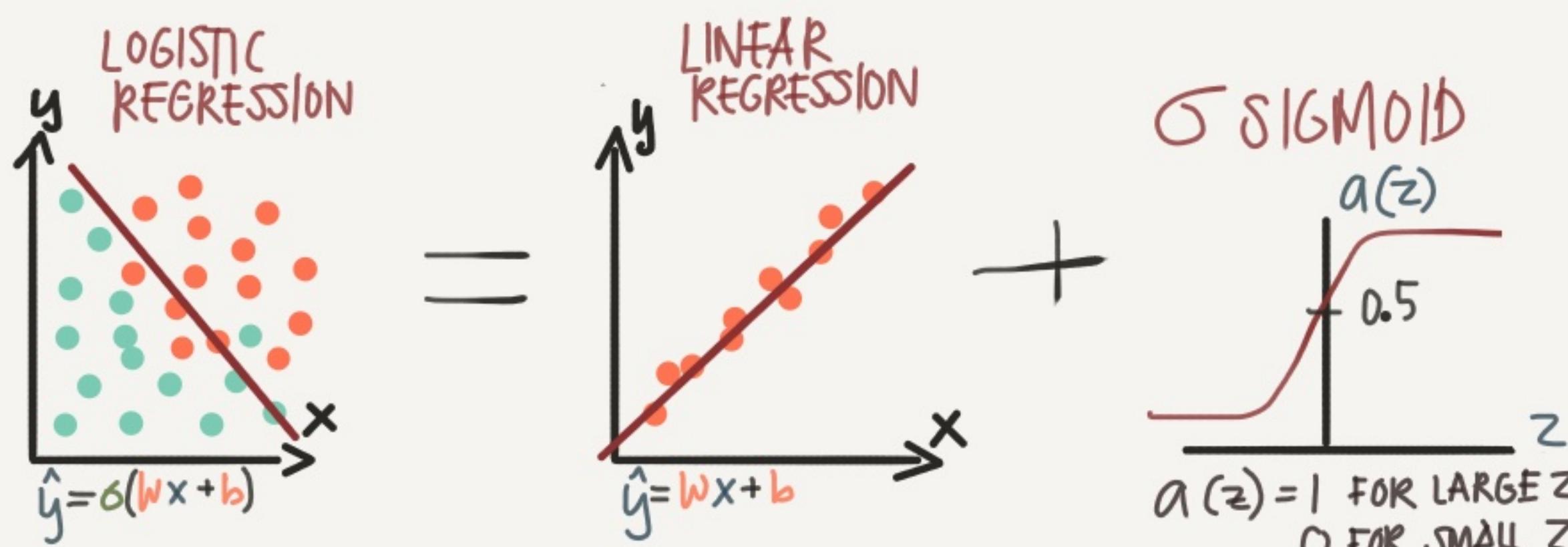
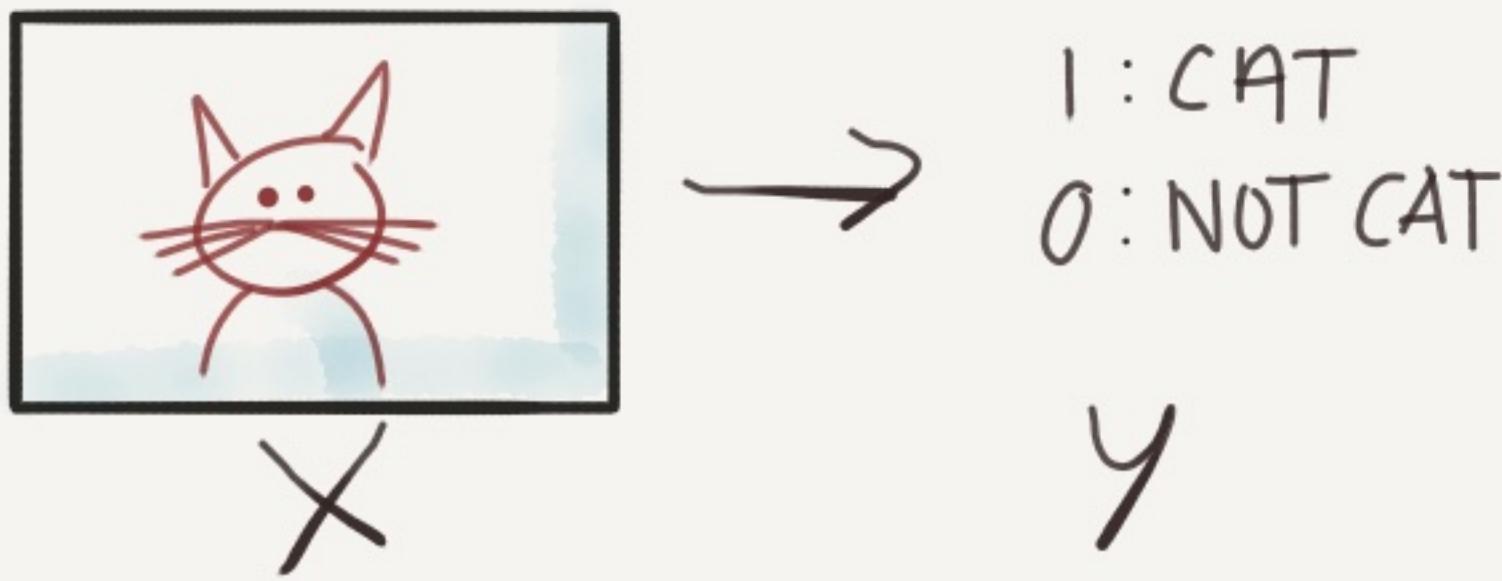
FASTER COMPUTATION  
IS IMPORTANT TO SPEED UP  
THE ITERATIVE PROCESS



SIGMOID

RELU

## BINARY CLASSIFICATION



THE TASK IS TO LEARN  $w$  &  $b$  BUT HOW?

A: OPTIMIZE HOW GOOD THE GUESS IS BY MINIMIZING THE DIFF BETWEEN GUESS ( $\hat{y}$ ) AND TRUTH ( $y$ )

$$\text{LOSS} = \mathcal{L}(\hat{y}, y)$$

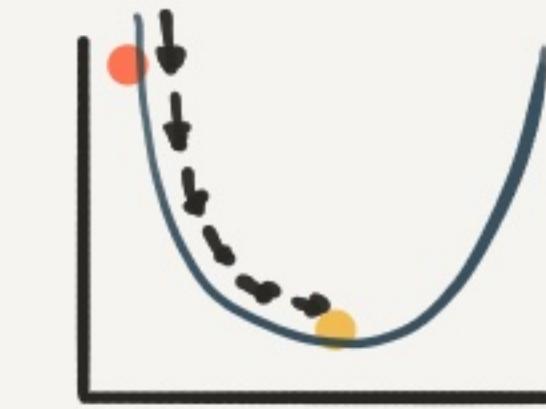
$$\text{COST} = J(w, b) = \frac{1}{m} \sum_{i=1}^m \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$$

COST = LOSS FOR THE ENTIRE DATASET

# LOGISTIC REGRESSION

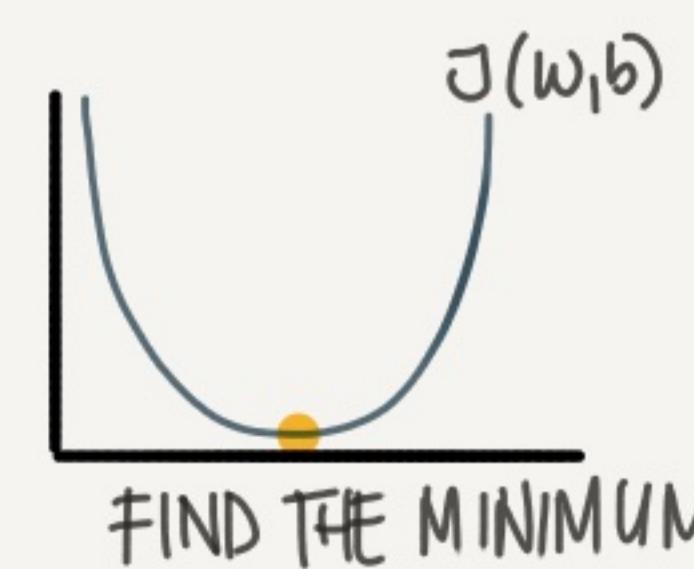
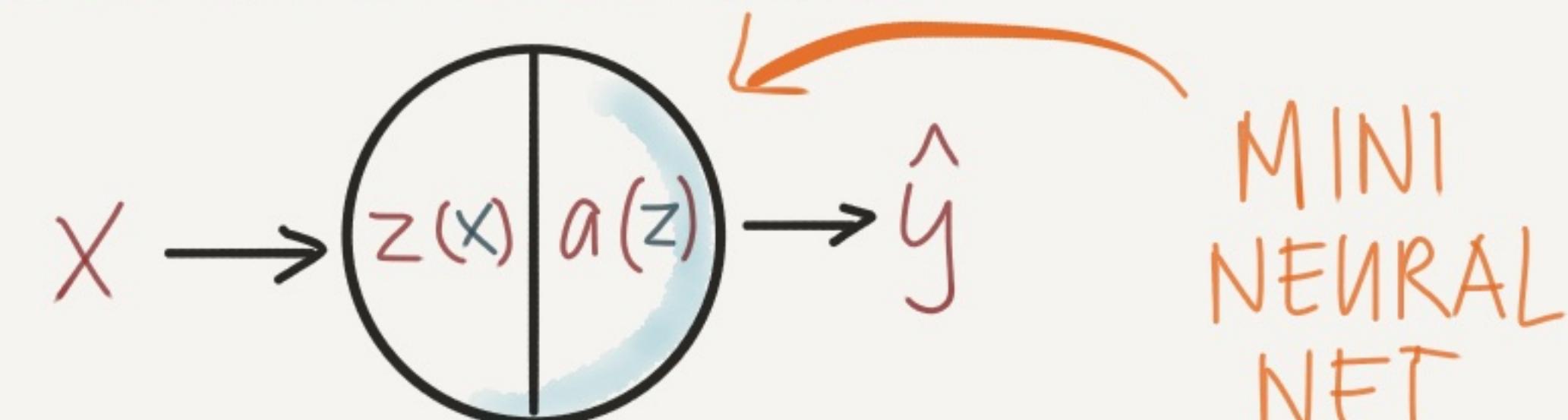
## AS A NEURAL NET

### FINDING THE MINIMUM WITH GRADIENT DESCENT



1. FIND THE DOWNSHILL DIRECTION (USING DERIVATIVES)
  2. WALK (UPDATE  $w$  &  $b$ ) AT A  $\alpha$  LEARNING RATE
- REPEAT UNTIL YOU REACH BOTTOM (CONVERGE)

### PUTTING IT ALL TOGETHER



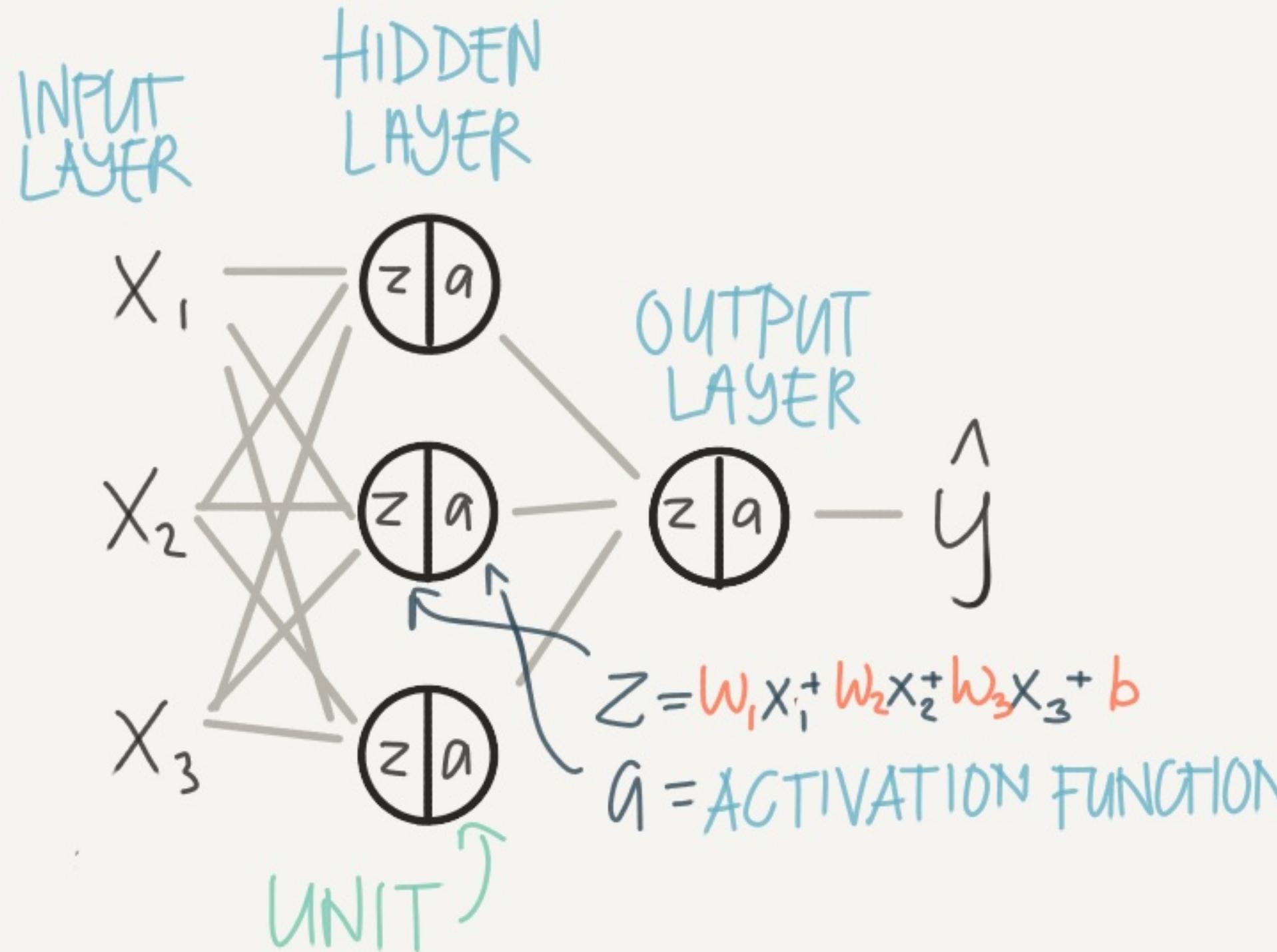
$$z(x) = w*x + b$$

$$\hat{y} = a(z) = \sigma \text{SIGMOID}(z)$$

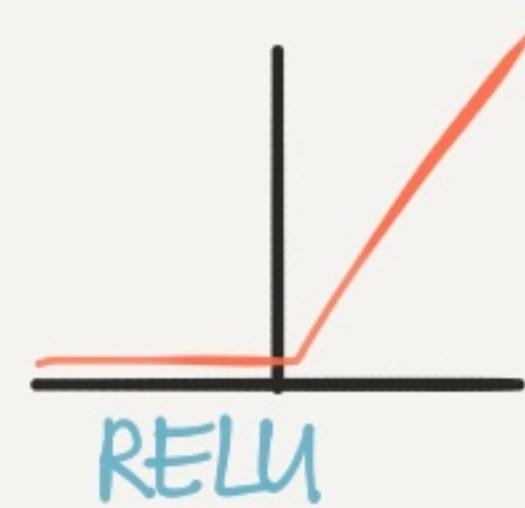
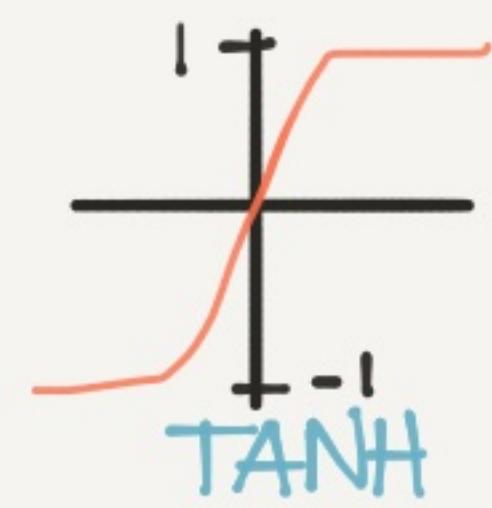
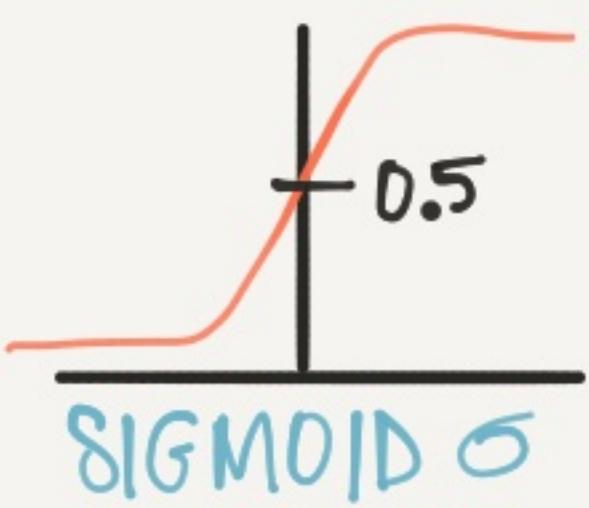
1. FORWARD PROPAGATION • CALCULATE  $\hat{y}$
2. BACKWARD PROPAGATION • GRADIENT DESCENT + UPDATE  $w$  &  $b$

REPEAT UNTIL IT CONVERGES

## 2 LAYER NEURAL NET



## ACTIVATION FUNCTIONS

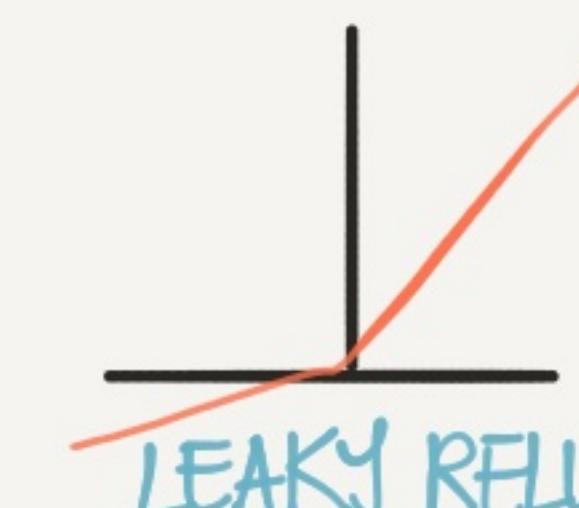


BINARY CLASSIFIER  
ONLY USED FOR  
OUTPUT LAYER

SLOW GRAD  
DESCENT SINCE  
SLOPE IS SMALL  
FOR LARGE/SMALL VAL

NORMALIZED  
 $\Rightarrow$  GRADIENT  
DESCENT IS  
FASTER

DEFAULT  
CHOICE FOR  
ACTIVATION  
SLOPE = 1/0



AVoids UNDEF  
SLOPE AT 0  
BUT RARELY  
USED IN PRACTICE

# SHALLOW NEURAL NETS

## WHY ACTIVATION FUNCTIONS?

EX. WITH NO ACTIVATION -  $a = z$

$$\begin{aligned} a^{[1]} &= z^{[1]} = W^{[1]}x + b^{[1]} \\ a^{[2]} &= z^{[2]} = W^{[2]}a^{[1]} + b^{[2]} \end{aligned}$$

LAYER 1  
LAYER 2

PLUG IN  $a^{[1]}$

$$\begin{aligned} a^{[2]} &= W^{[2]}(W^{[1]}x + b^{[1]}) + b^{[2]} \\ &= \underbrace{W^{[2]}W^{[1]}x}_{W'x} + \underbrace{W^{[2]}b^{[1]} + b^{[2]}}_{b'} \end{aligned}$$

LINEAR  
FUNCTION

## INITIALIZING $W+b$

WHAT IF: INIT TO  $\emptyset$

THIS WILL CAUSE ALL THE UNITS  
TO BE THE SAME AND LEARN  
EXACTLY THE SAME FEATURES

SOLUTION: RANDOM INIT  
BUT ALSO WANT THEM  
SMALL SD RAND  $\approx 0.01$

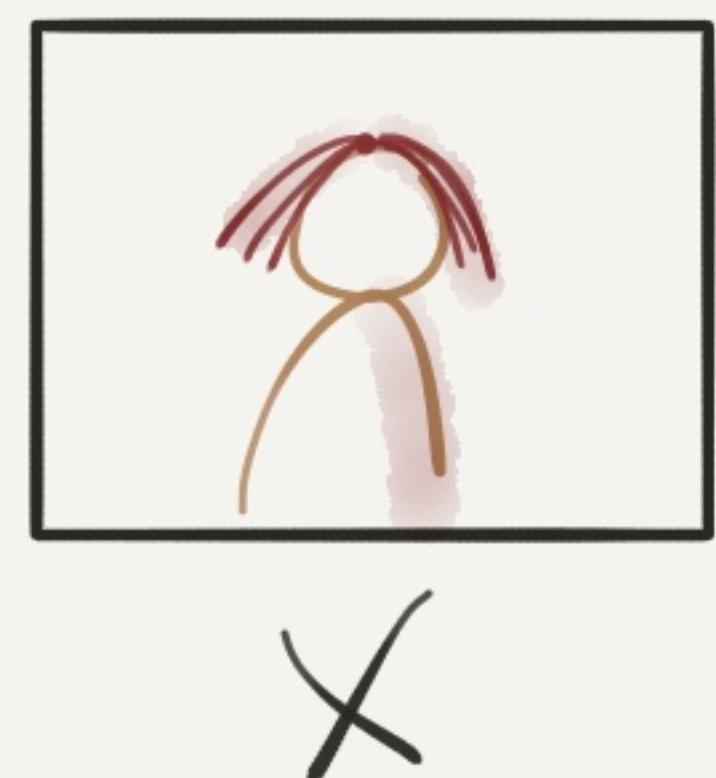
WE COULD JUST  
AS WELL HAVE  
SKIPPED THE WHOLE  
NEURAL NET &  
USED LIN. REGR.

HYPERPARAM  
© Tess Ferrandez

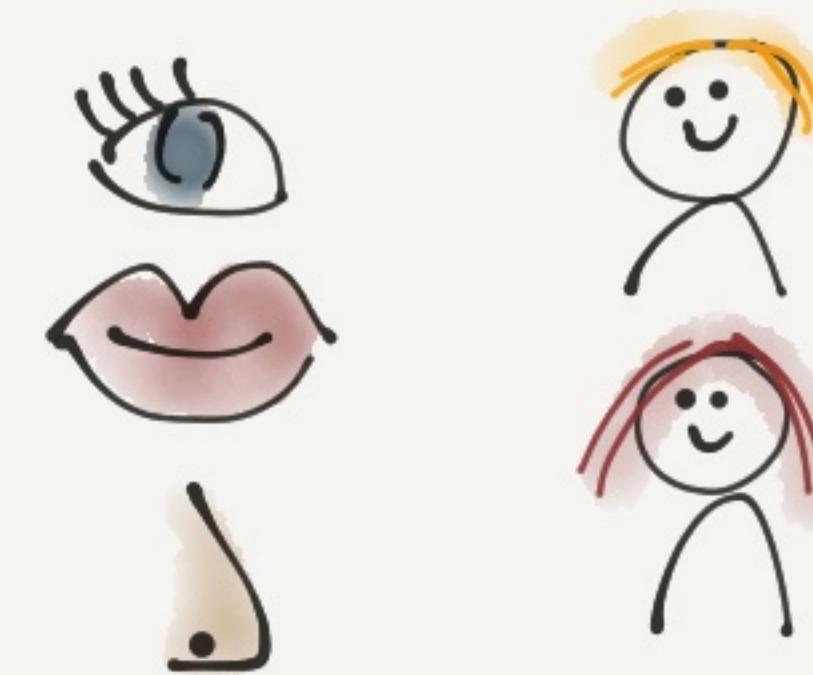
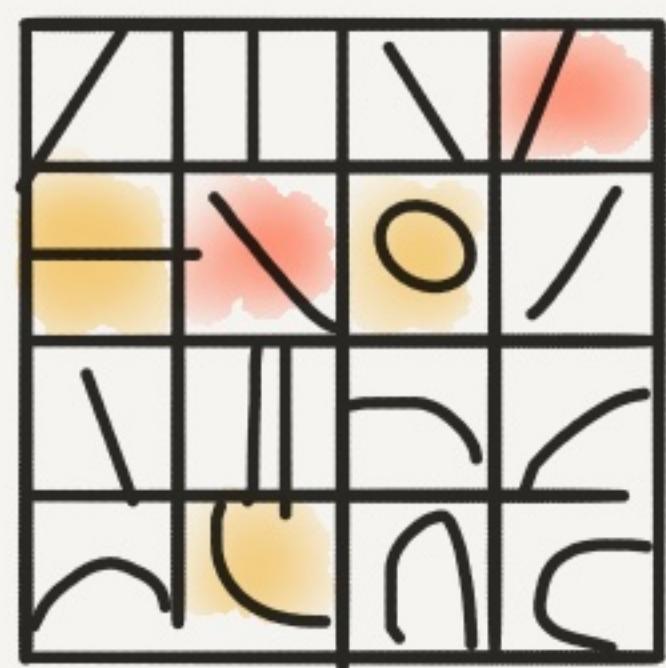
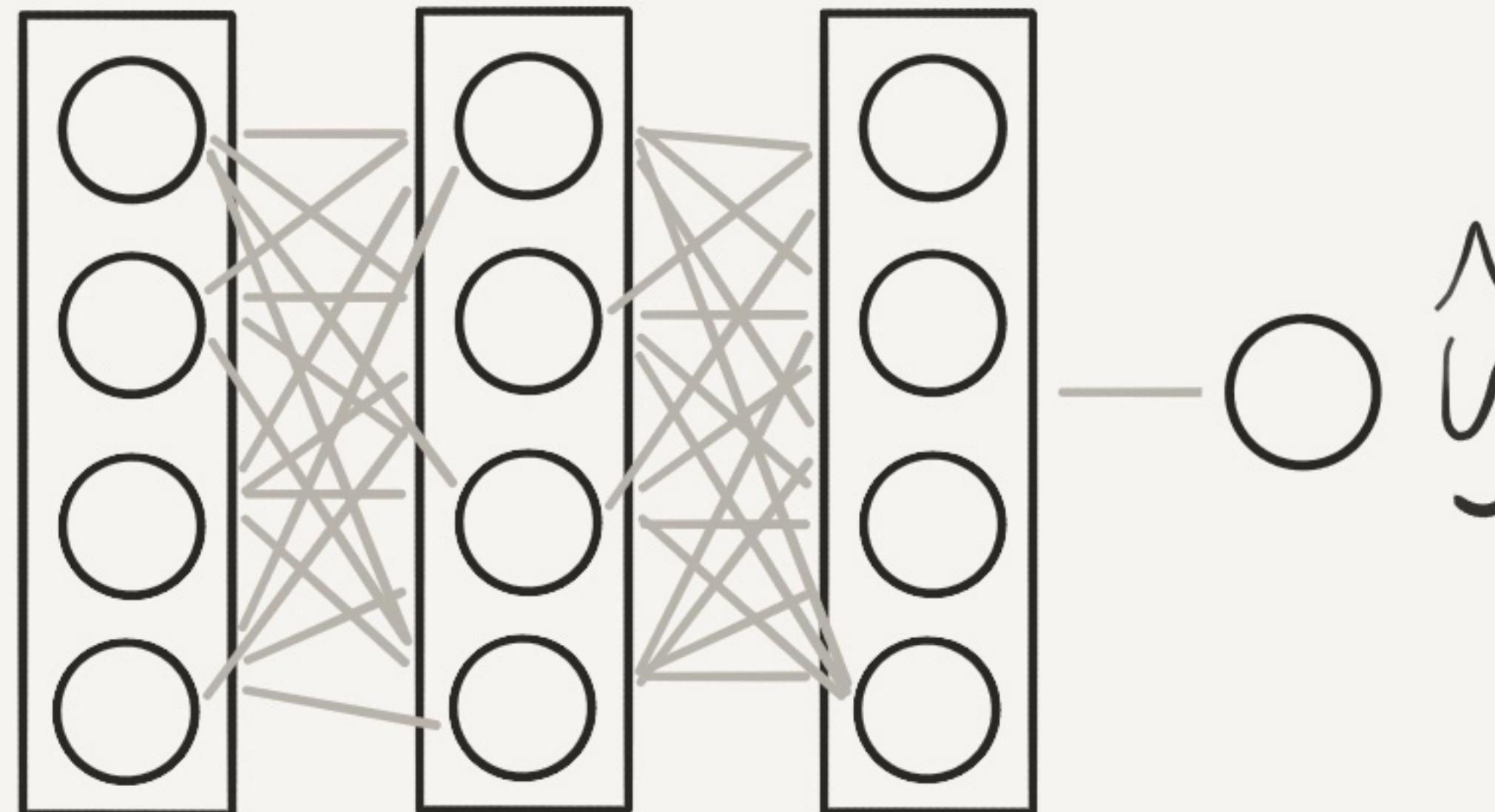
# DEEP NEURAL NETS

WHY DEEP NEURAL NETS?

THERE ARE FUNCTIONS A  
SMALL DEEP NET CAN COMPUTE  
THAT SHALLOW NETS NEED EXP.  
MORE UNITS TO COMP.



X



LOW LEVEL  
AUDIO WAVE  
FEATURES  
↑ ↓ PITCH



— PHONEMES — WORDS — SENTENCES

C A T

VERY DATA HUNGRY

NEED LOTS OF COMPUTER  
POWER

ALWAYS VECTORIZE  
VECTOR MULT. CHEAPER THAN FOR LOOPS

COMPUTE ON GPUs

LOTS OF HYPERPARAMS

- LEARNING RATE  $\alpha$
- # HIDDEN UNITS
- # ITERATIONS
- # HIDDEN LAYERS
- CHOICE OF ACTIVATION
- MOMENTUM
- MINI-BATCH SIZE
- REGULARIZATION

# SETTING UP YOUR ML APP

## CLASSIC ML

100 - 1000 SAMPLES

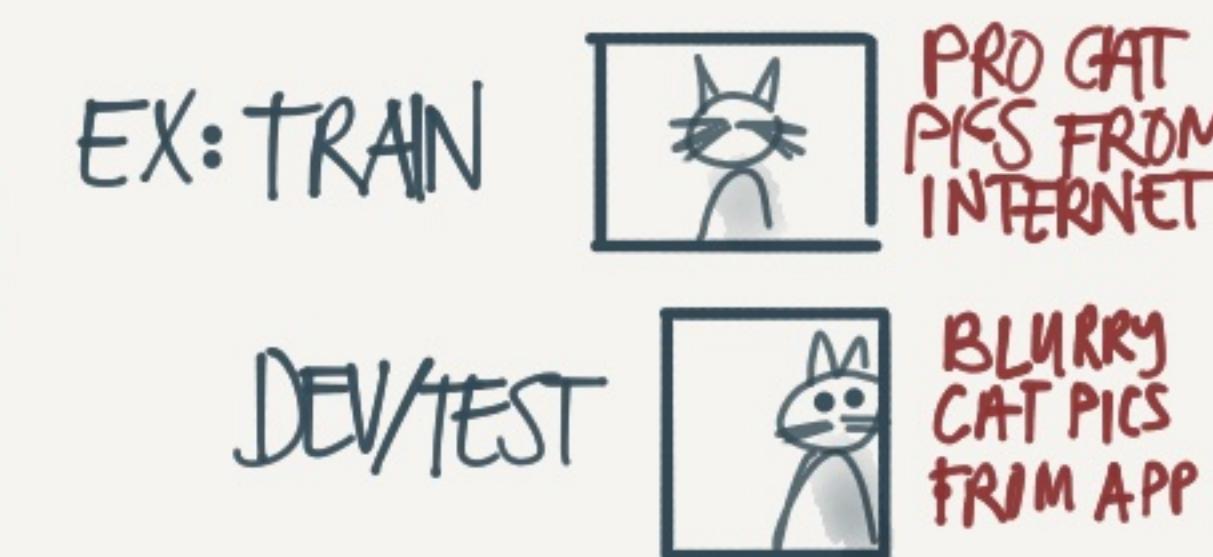
TRAIN	DEV	TEST
60%	20%	20%

ALL FROM SAME PLACE  
DISTRIBUTION

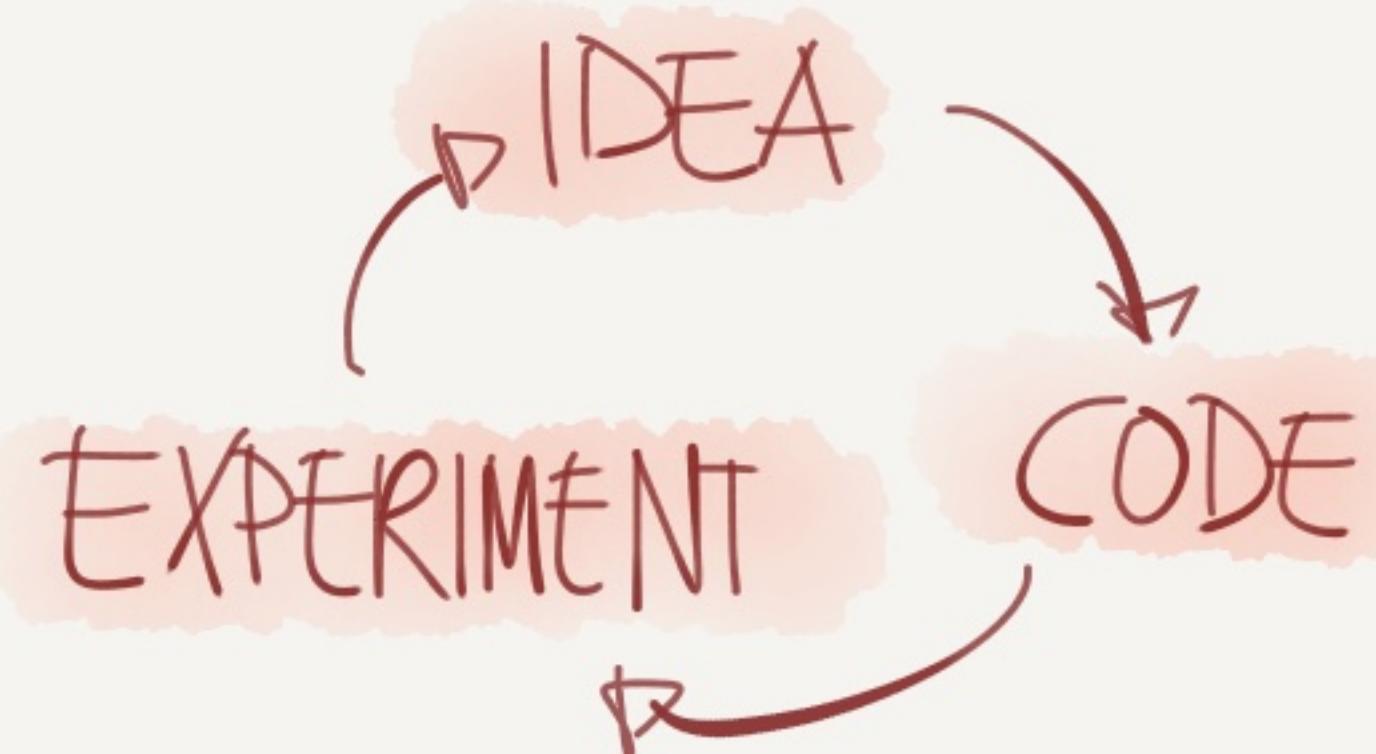
## DEEP LEARNING

1M SAMPLES

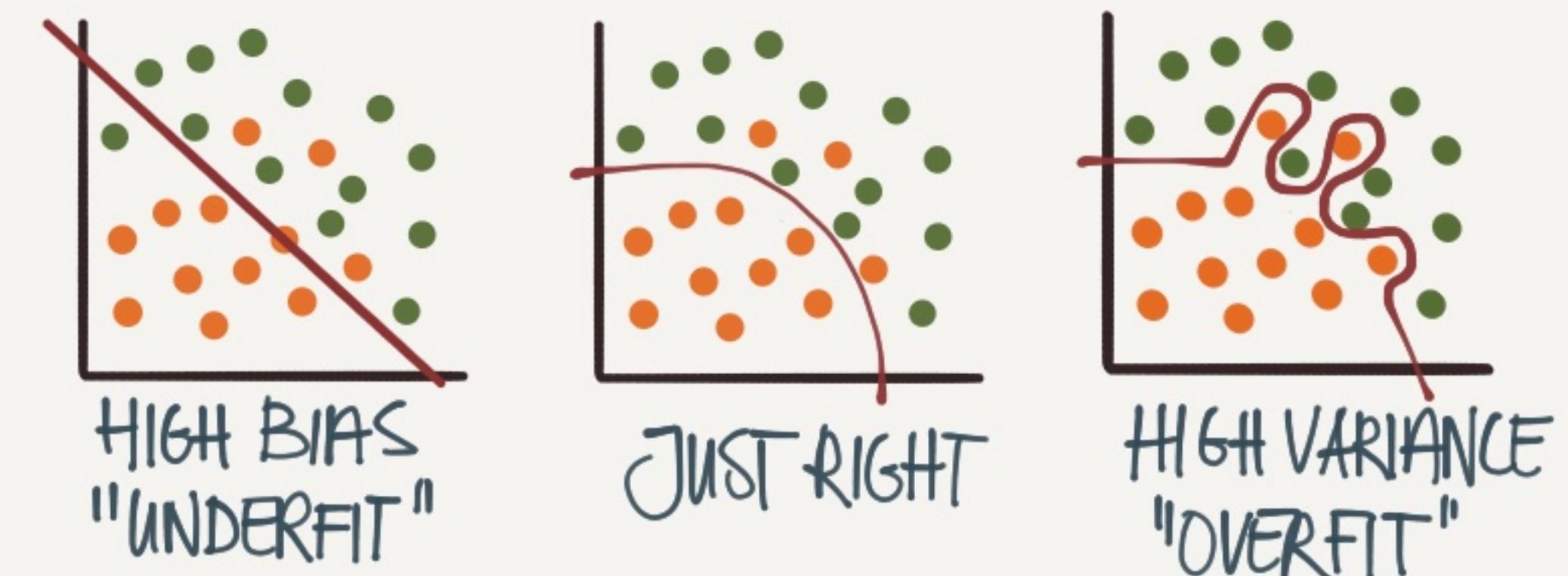
TRAIN	D	T
98%	1%	1%



 TIP  
DEV & TEST SHOULD COME  
FROM SAME DISTRIBUTION



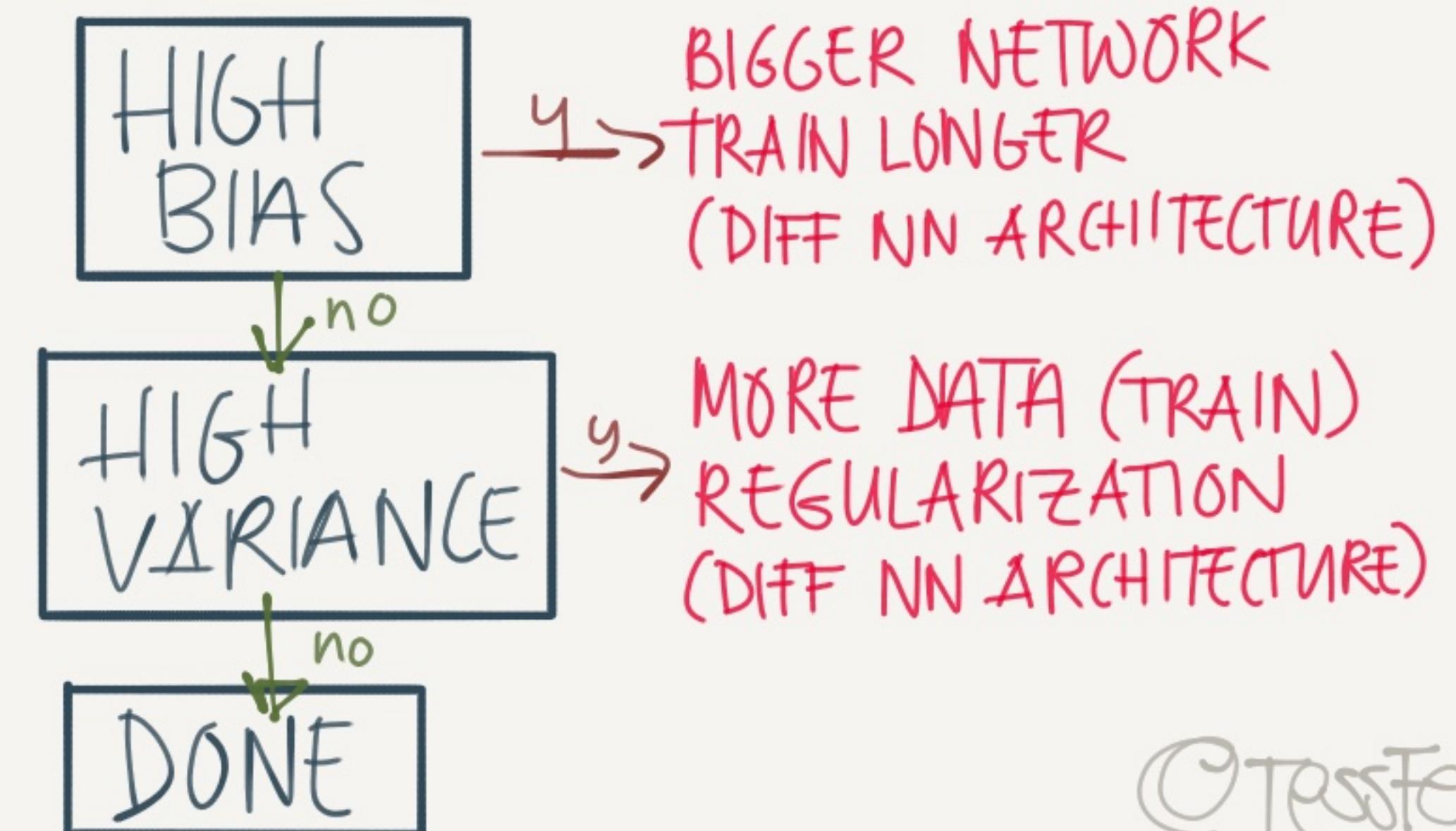
## BIAS / VARIANCE



		ERROR			
		1%	15%	15%	0.5%
TRAIN	1%	15%	15%	0.5%	
TEST	11%	16%	30%	1%	
	HIGH VARIANCE	HIGH BIAS	HIGH BIAS & VARIANCE	LOW BIAS & VARIANCE	

ASSUMING HUMANS GET 0% ERROR

## THE ML RECIPE



# REGULARIZATION

## PREVENTING OVERFITTING

### L2 REGULARIZATION

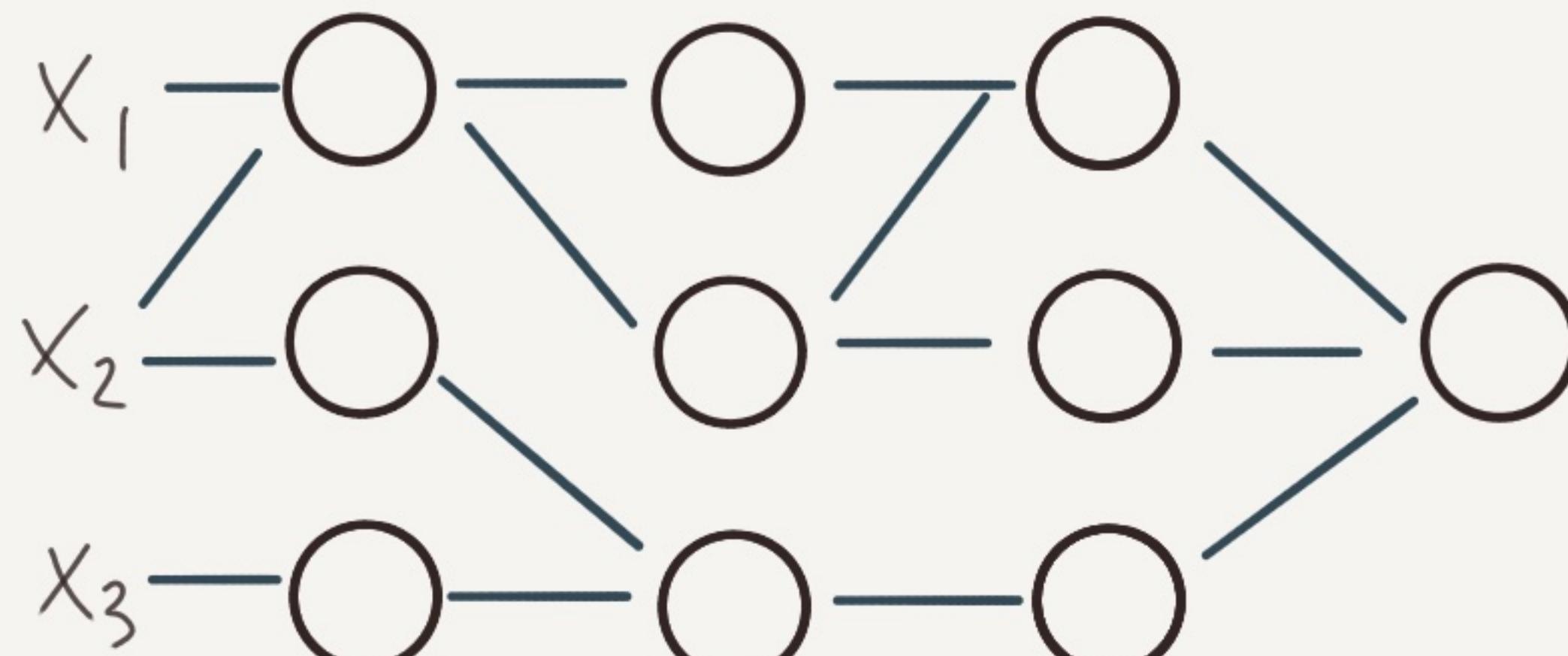
$$\text{COST: } J(w, b) = \frac{1}{m} \sum_{i=1}^m \ell(\hat{y}_i, y_i) + \frac{\lambda}{2m} \|w\|_2^2$$

EUCLIDEAN NORM

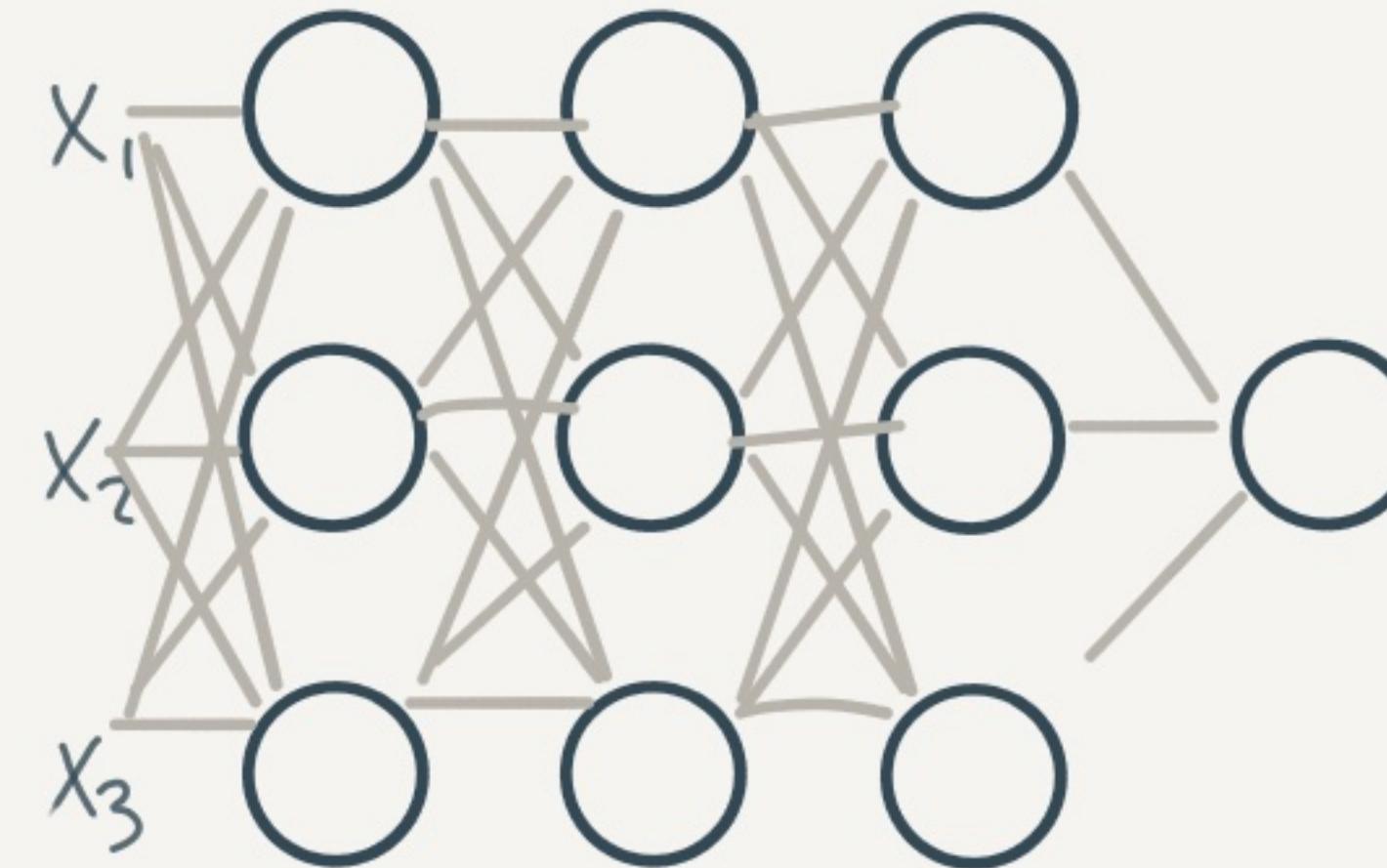
### L1 REGULARIZATION

$$\text{COST: } J(w, b) = \frac{1}{m} \sum_{i=1}^m \ell(\hat{y}_i, y_i) + \frac{\lambda}{m} \|w\|_1$$

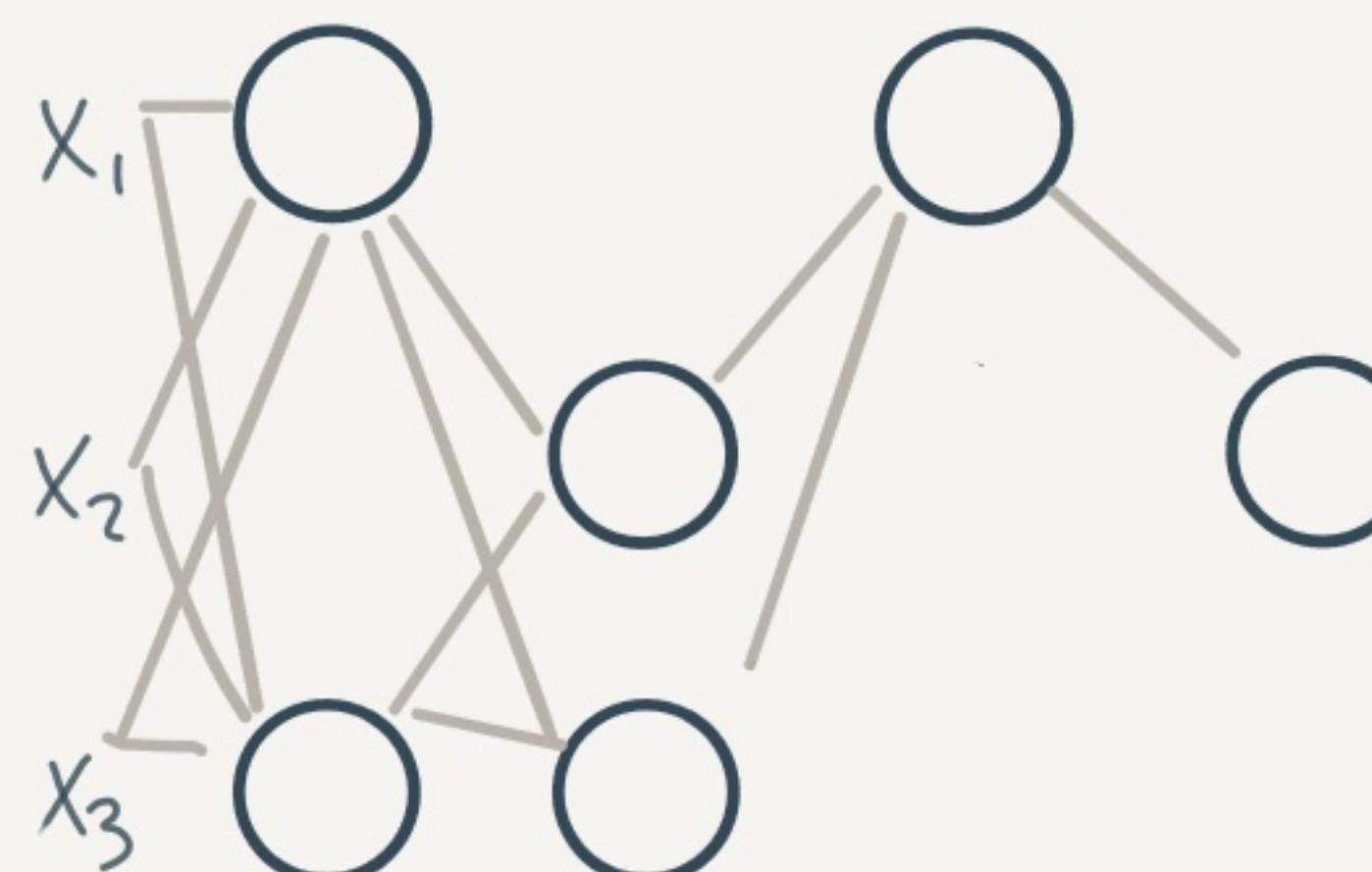
BOTH PENALIZE LARGE WEIGHTS  $\Rightarrow$   
 SOME WILL BE CLOSE TO  $0 \Rightarrow$   
 SIMPLER NETWORKS



### DROPOUT



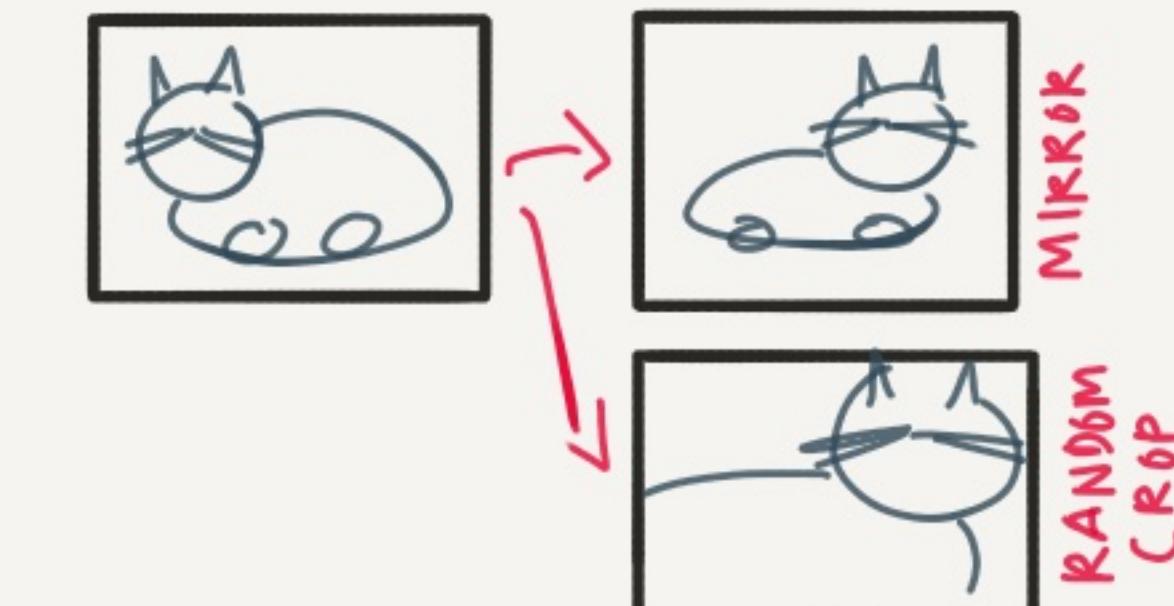
FOR EACH ITERATION  $i$  SAMPLE  
 SOME NODES ARE RANDOMLY  
 DROPPED (BASED ON KEEP-PROB)



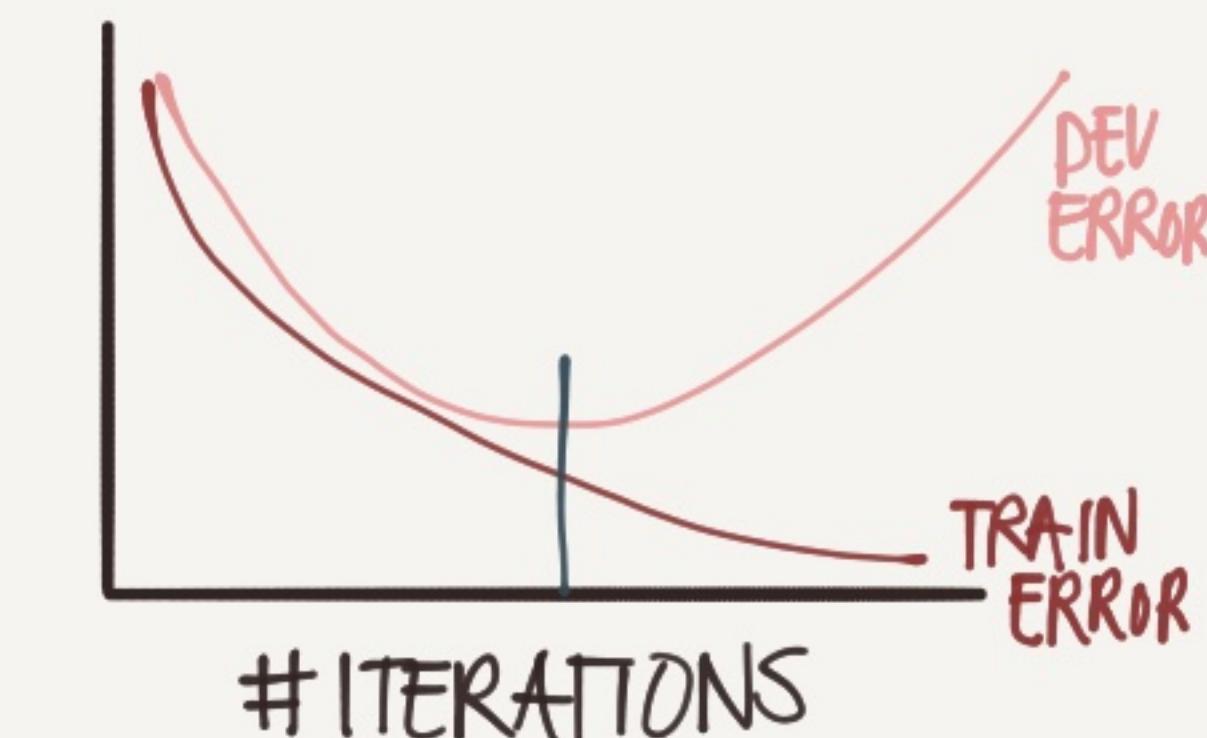
WE GET SIMPLER NWs  
 & LESS CHANCE TO RELY ON  
 SINGLE FEATURES

### OTHER REGULARIZATION TECHNIQUES

DATA AUGMENTATION  
 GENERATE NEW PICS FROM EXISTING



### EARLY STOPPING

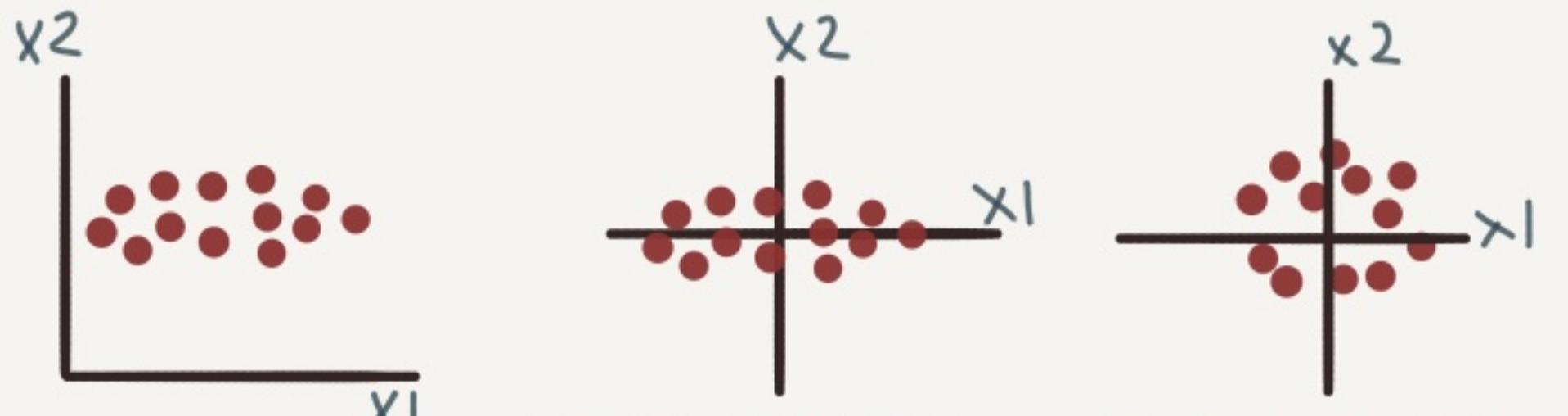


PROBLEM: AFFECTS BOTH  
 BIAS & VARIANCE

# OPTIMIZING

## TRAINING

### NORMALIZING INPUTS



STEP1: CENTER  
AROUND 0,0

STEP2: SCALE  
SO VARIANCE IS SAME  
 $Cx - 1 \rightarrow 1$

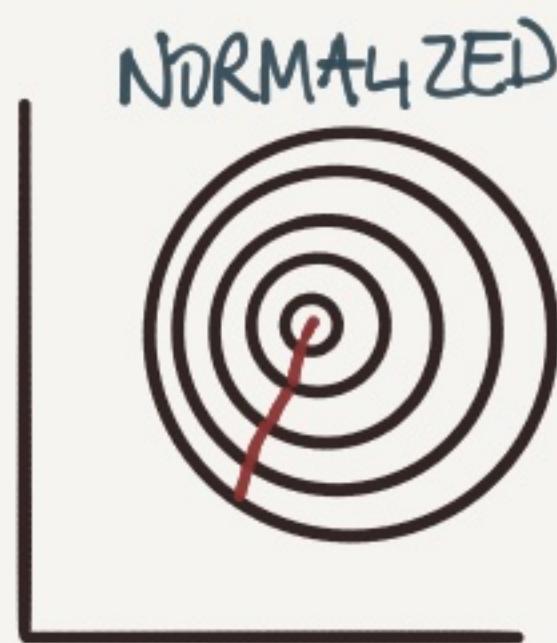


TIP  
USE SAME AVG/VAR TO  
NORMALIZE DEV/TEST

### WHY DO WE DO THIS?



IF WE NORMALIZE, WE CAN USE A MUCH  
LARGER LEARNING RATE  $\alpha$



### DEALING WITH VANISHING/EXPLODING GRADIENTS

Ex: DEEP NW (L LAYERS)

$$\hat{y} = \underbrace{w^{[L-1]} w^{[L-2]} \cdots w^{[0]}}_{w} x + b$$

IF  $w = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \Rightarrow 0.5^{L-1} \Rightarrow$  VANISHING

OR  $w = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix} \Rightarrow 1.5^{L-1} \Rightarrow$  EXPLODING

IN BOTH CASES GRADIENT DESCENT  
TAKES A VERY LONG TIME

PARTIAL SOLUTION: CHOOSE INITIAL  
VALUES CAREFULLY

$$w^{[l]} = \text{rand} * \sqrt{\frac{2}{n^{l-1}}} \quad (\text{FOR RELU})$$

$$\# \text{inputs} \quad (\text{FOR TANH})$$

$$\sqrt{\frac{1}{n^{l-1}}} \quad (\text{FOR TANH})$$

SETS THE VARIANCE

### GRADIENT CHECKING

IF YOUR COST DOES NOT  
DECREASE ON EACH ITER  
YOU MAY HAVE A  
BACKPROP BUG.

GRADIENT CHECKING  
APPROXIMATES THE  
GRADIENTS SO YOU  
CAN VERIFY CALC.

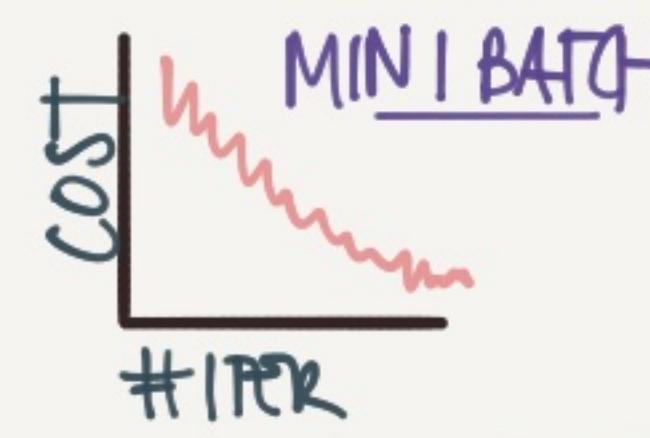
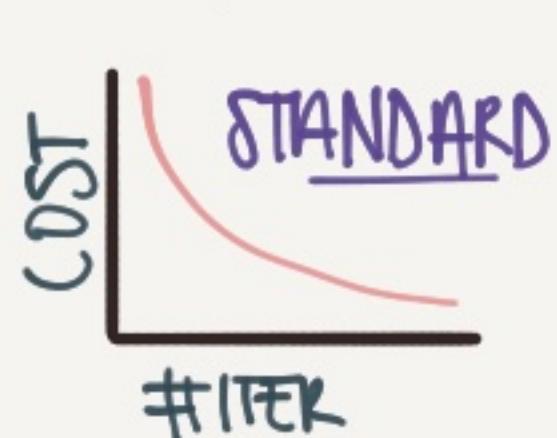
NOTE ONLY USE  
WHEN DEBUGGING  
SINCE IT'S SLOW

# OPTIMIZATION ALGORITHMS

## MINI-BATCH GRAD. DESCENT

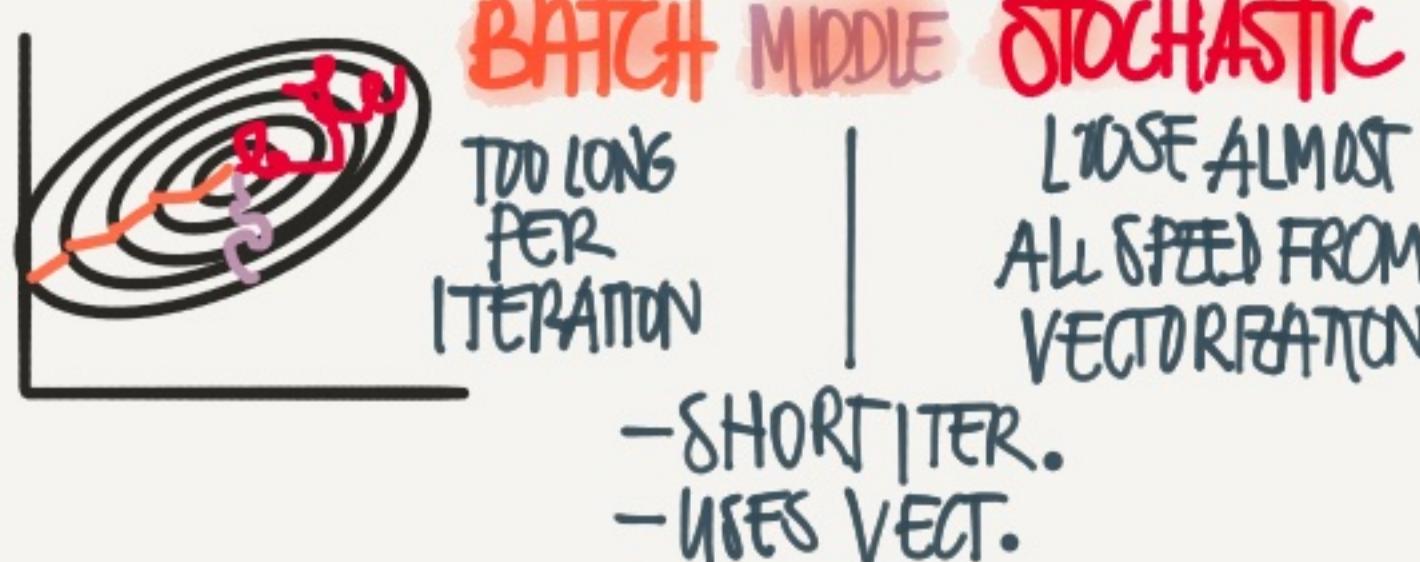


SPLIT YOUR DATA INTO MINI-BATCHES & DO GRAD DESCENT AFTER EACH BATCH THIS WAY YOU CAN PROGRESS AFTER JUST A SHORT WHILE



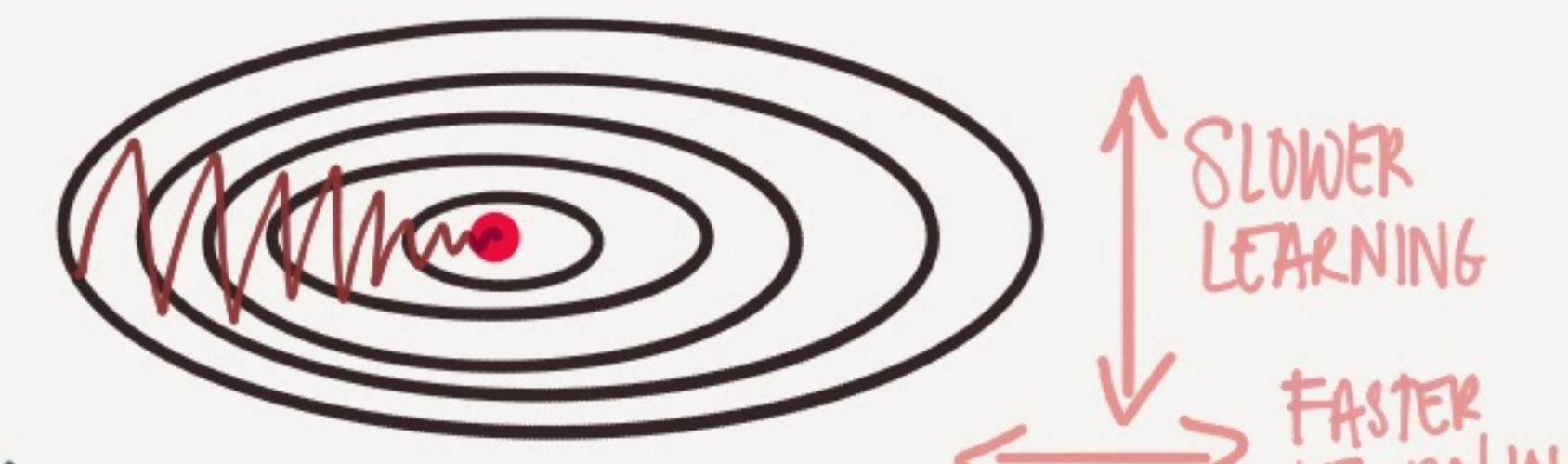
## CHOOSING THE MINIBATCH SIZE

$\delta \text{SIZE} = m \rightarrow$  BATCH GRAD DESC.  
 $\delta \text{SIZE} = 1 \rightarrow$  STOCHASTIC GRAD DESC



**TIP**  
 IF YOU HAVE  $< 2000$  SAMPLES  
 USE  $\text{SIZE}=2000$   
 OTHERWISE, USE 64, 128, 256...  
 SO X+Y FITS IN CPU/GPU CACHE

## GRADIENT DESCENT W. MOMENTUM



WE WANT TO REDUCE OSCILLATION  $\updownarrow$  SO WE GET TO THE GOAL FASTER

**SOLUTION:** SMOOTH OUT THE CURVE BY TAKING AN EXPONENTIALLY WEIGHTED AVERAGE OF THE DERIVATIVES (i.e. LAST ONE HAS MORE IMPORTANCE)

## RMSProp - ROOT MEAN SQUARED



NORMALIZE GRADIENT USING A MOVING AVG.

$$S_{dw} = \beta S_{dw} + (1-\beta) dw^2$$

$$S_{db} = \beta S_{db} + (1-\beta) db^2$$

$$w = w - \alpha \frac{dw}{\sqrt{S_{dw}}} \quad b = b - \alpha \frac{db}{\sqrt{S_{db}}}$$

## ADAM OPTIMIZATION

COMBO OF GD w/ MOMENTUM & RMSProp

## LEARNING RATE DECAY

IDEA: USE A LARGE  $\alpha$  IN THE BEGINNING THEN DECREASE AS WE GET CLOSER TO GOAL

$$\text{OPTION 1: } \alpha = \frac{1}{1 + \text{DECAYRATE} \cdot \text{EPOCH}} \alpha_0$$

$$\text{EXponential: } \alpha = 0.95^{\text{EPOCH}} \alpha_0$$

$$\text{OPTION 3: } \alpha = \frac{k}{\sqrt{\text{EPOCH}}} \alpha_0$$

$$\text{OPTION 4: } \alpha = \frac{k}{\sqrt{t}} \alpha_0$$

$$\text{OPTION 5: DISCRETE STAIRCASE}$$

$$\text{OPTION 6: MANUAL}$$

EPOCH = 1 PASS THROUGH THE DATA

# HYPERPARAM TUNING

WHICH HYPERPARAMS ARE MOST IMPORTANT?

$\alpha$  LEARNING RATE

# HIDDEN UNITS

MINIBATCH SIZE

$\beta$  MOMENTUM, TURN = 0.9

# LAYERS

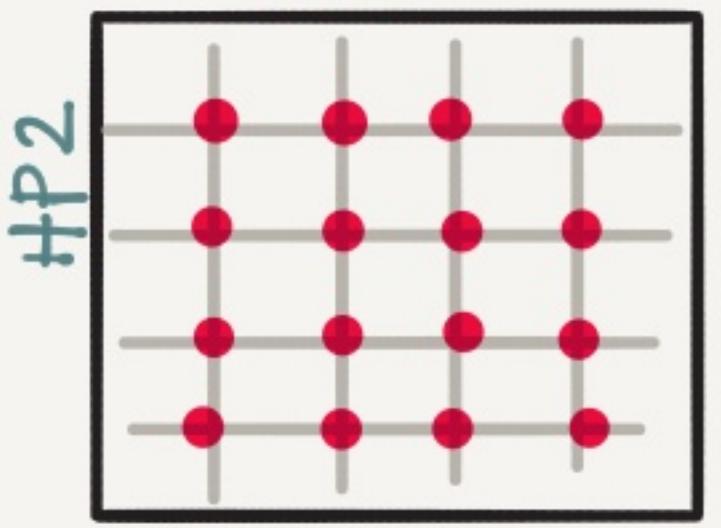
LEARNING RATE DECAY

$\beta_1 = 0.9 \quad \beta_2 = 0.999 \quad \epsilon = 10^{-8}$  (ADAM)

## TESTING VALUES

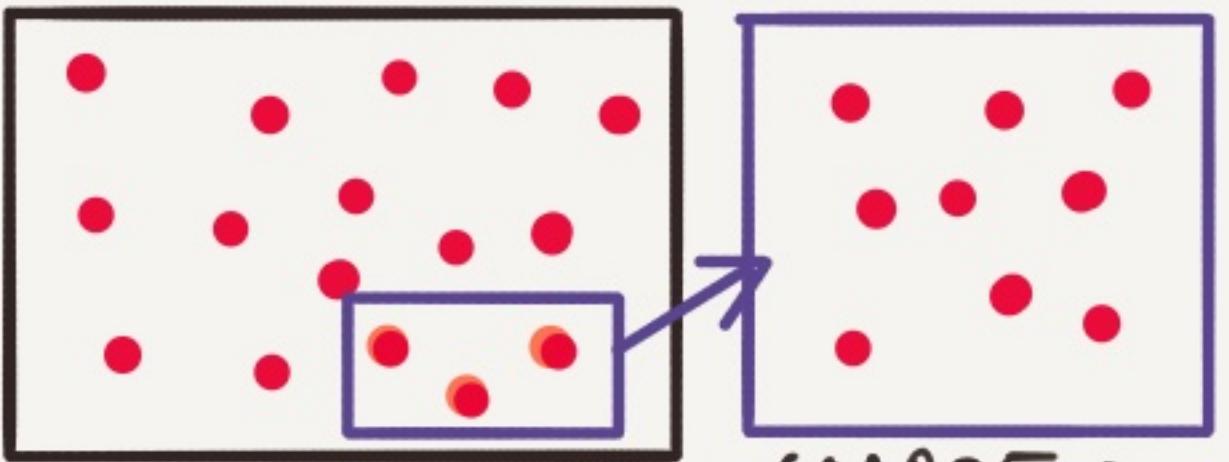
### CLASSIC ML

HP1



GRID SEARCH

### SOLUTION



RANDOM SEARCH + COARSE  $\rightarrow$  DENSE

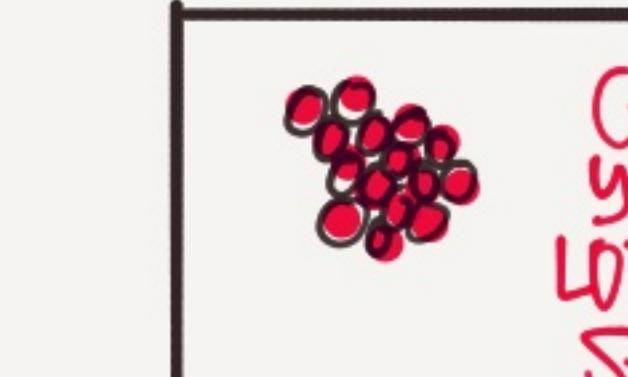
PROBLEM: ONE ITERATION TAKES A LONG TIME & IN 16 GO'S WE HAVE ONLY TRIED 4 $\alpha$  - BUT 4 DIFF  $\epsilon$

NOT AS IMPORTANT

MY PANDA IS ACTUALLY A MIS-CATEGORIZED CAT BECAUSE I CAN'T DRAW PANDAS



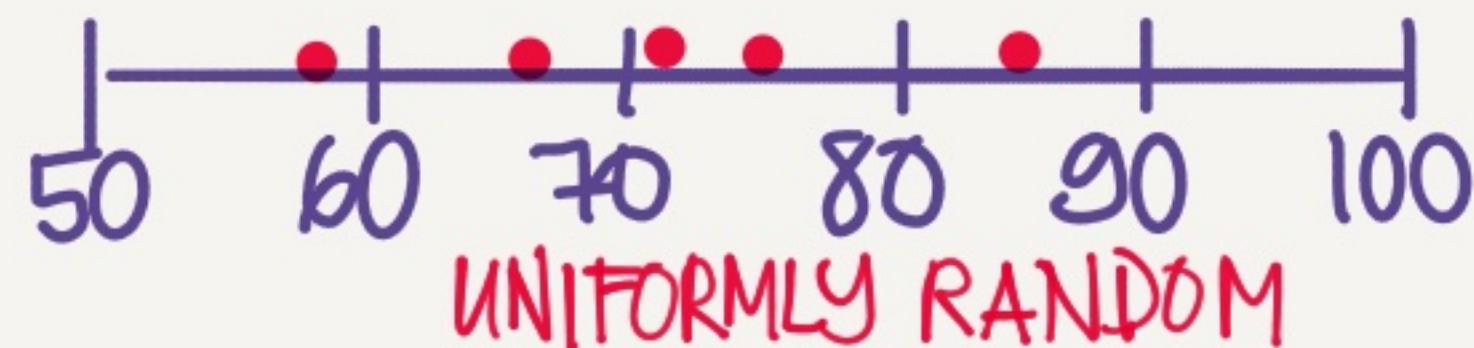
BABYSIT ONE MODEL & TUNE



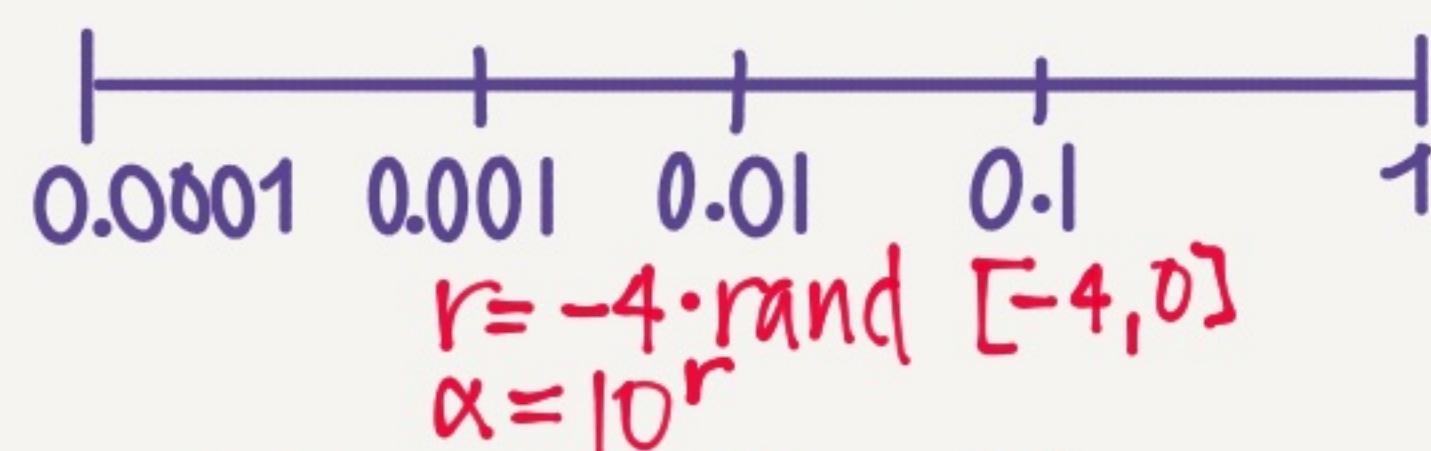
SPAWN LOTS OF MODELS W DIFF HP

## USE AN APPROPRIATE SCALE

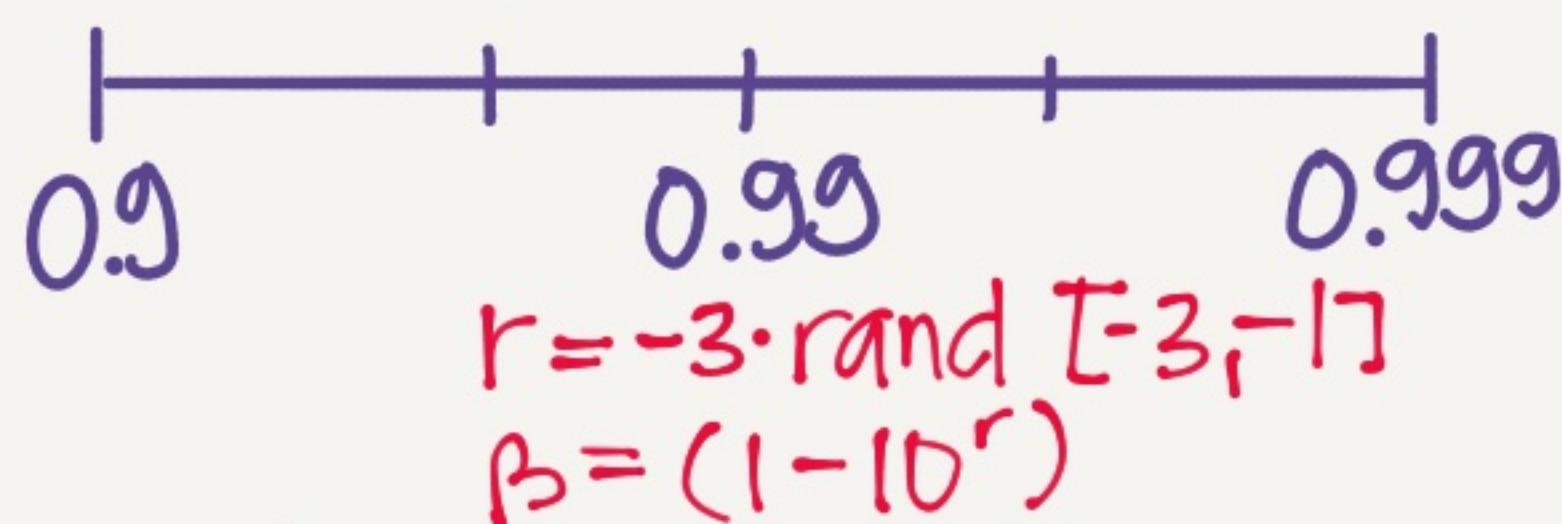
### # HIDDEN UNITS



### $\alpha$ LEARNING RATE



### $\beta$ EXP WEIGHT AVE



TIP  
RE-EVALUATE YOUR HYP. PARAMS EVERY FEW MONTHS

### PANDA VS CAVIAR

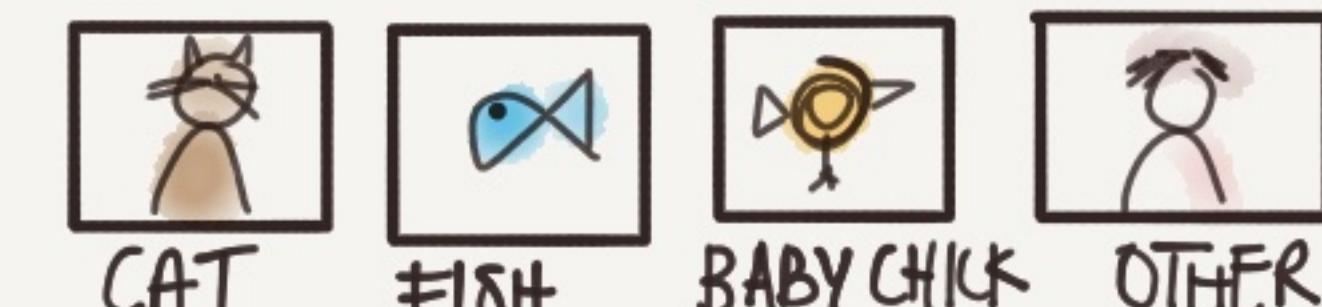
## MISC. EXTRAS

### BATCH NORMALIZATION

#### NORMALIZE LAYER OUTPUT

- SPEEDS UP TRAINING
- MAKES WEIGHTS DEEPER IN NW MORE ROBUST (COVARIATE SHIFT)
- SIGHT REGULARIZING EFFECT

### MULTICLASS CLASSIFIC.

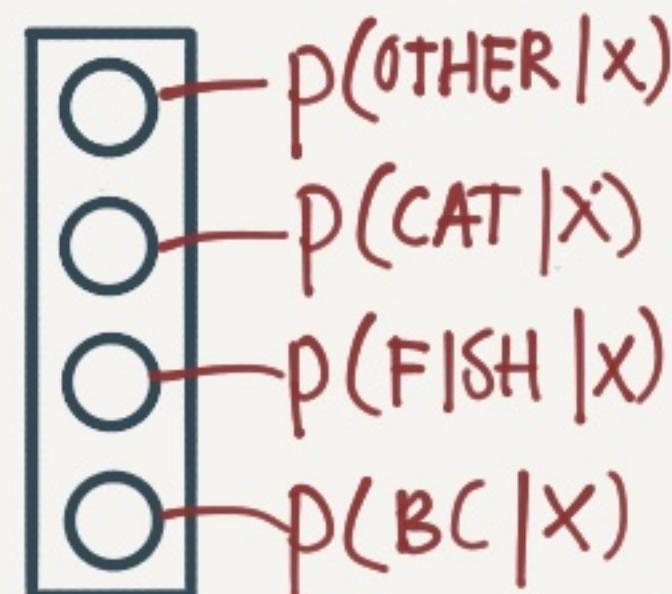


C = # CLASSES = 4

### SOFTMAX ACTIVATION

$$t = e^{(z^{[i]})}$$

$$a^{[i]} = \frac{t}{\sum t_i}$$



EX:  $z^{[i]} = \begin{bmatrix} 5 \\ 2 \\ -1 \\ 3 \end{bmatrix}$   $t = \begin{bmatrix} e^5 \\ e^2 \\ e^{-1} \\ e^3 \end{bmatrix} = \begin{bmatrix} 148.4 \\ 7.4 \\ 0.4 \\ 20.1 \end{bmatrix}$

$$\Rightarrow a^{[i]} = \frac{t}{176.3} = \begin{bmatrix} 0.842 \\ 0.042 \\ 0.02 \\ 0.114 \end{bmatrix}$$

11.4% PROB IT'S A BABY CHICK

© TessFerrandez

# STRUCTURING YOUR ML PROJECTS

## SETTING YOUR GOAL

\* GOAL SHOULD BE A SINGLE #

	PRECISION	RECALL	
A	95%	90%	IS A OR B BEST?
B	98%	85%	

	PRECISION	RECALL	F1
A	95%	90%	92.4%
B	98%	85%	91%

F1 = HARMONIC MEAN BETW.  
RECALL & PRECISION

\* DEFINE OPTIMIZING VS  
SATISFYING METRICS

	ACCURACY	RUNTIME
A	90%	80ms
B	92%	95ms
C	95%	1500ms

MAXIMIZE ACC.  
GIVEN TIME < 100ms

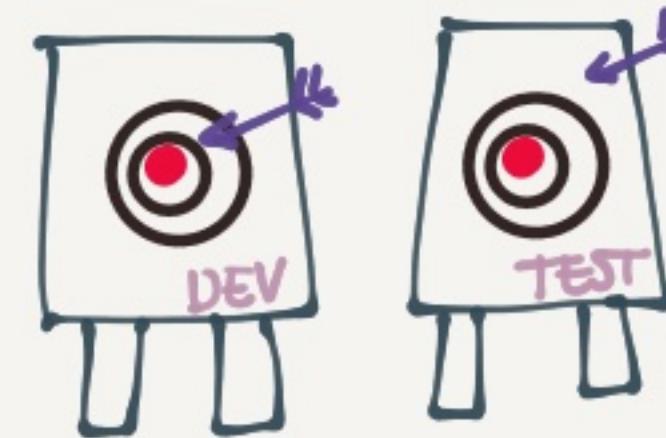
ACCURACY =  
OPTIMIZING  
RUNTIME =  
SATISFYING

## SELECTING YOUR DEV/TEST SETS

### DATA

US  
UK  
EUROPE  
S.AM  
INDIA  
CHINA  
AUST.

OPTION 1:  
DEV = UK, US, EUR  
TEST = REST



IF DEV & TEST ARE DIFF  
& WE OPTIMIZE FOR DEV  
WE WILL MISS THE TEST TARGET

## HUMAN LEVEL PERF



WHY DOES ACC  
SLOW DOWN WHEN  
WE SURPASS HUMAN  
LEVEL PERF?

MEDICAL IM6 CLASS  
TYPICAL HUMAN 3%  
TYPICAL DOCTOR 1%  
EXPERIENCED DR. 0.7%  
TEAM OF EXP DRs. 0.5%

↑ HUMAN LEV PERF  
(PROXY FOR BAYES)

- OFTEN CLOSE TO BAYES
- A HUMAN CAN NO LONGER  
HELP IMPROVE (INSIGHTS)
- DIFFICULT TO ANALYSE  
BIAS/VARIANCE

## CAT CLASSIFICATION

	A	B	BLURRY
HUMAN	1%	7.5%	
TRAIN ERR	8%	8%	AVOIDABLE BIAS
DEV ERR	10%	10%	VARIANCE

FOCUS ON BIAS      FOCUS ON VARIANCE

HUMAN TRAIN BIGGER NETW.  
| AVOIDABLE BIAS } TRAIN LONGER/BETTER OPT. (RMSprop, ADAM)  
TRAIN | ALSO CHANGE NN ARCH OR HYPERPARAMS  
| VARIANCE } MORE DATA (TRAIN)  
DEV } REGULARIZATION NN ARCHITECTURE

	A	B	
HUMAN	0.5	0.5	AVOIDABLE BIAS
TRAIN ERR	0.6	0.3	VARIANCE
DEV ERR	0.8	0.4	
AVOID. BIAS	0.1	?	DON'T KNOW IF WE OVERFIT OR IF WE'RE CLOSE TO BAYES

OPTIONS TO PROCEED ARE UNCLEAR

# ERROR ANALYSIS

YOU HAVE 10% ERRORS, SOME ARE DOGS MIS-CLASSIFIED AS CATS. SHOULD YOU TRAIN ON MORE DOG PICS?

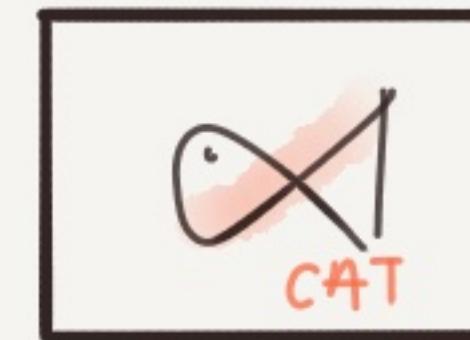
1. PICK 100 MIS-LABLED
2. COUNT ERROR REASONS

Dog	Blurry	Insta Filter	Big Cat	...
1	1		1	
2			1	
3		1		
...				
100			1	
5	...			

5% OF ALL ERRORS

FOCUSING ON DOGS. THE BEST WE CAN HOPE FOR IS 9.5% ERROR

YOU FIND SOME INCORR. LABELED DATA IN THE DEV SET. SHOULD YOU FIX IT?



DL ALGORITHMS ARE PRETTY ROBUST TO RANDOM ERRORS. BUT NOT TO SYSTEMATIC ERR.  
(EX. ALL WHITE CATS INCORR LABLED AS MICE)

ADD EXTRA COL. IN ERROR ANALYSIS AND USE SAME CRITERIA

**NOTE** IF YOU FIX DEV YOU SHOULD FIX TEST AS WELL.

FOR NEW PROJ. ·  
BUILD 1ST SYSTEM QUICK & ITERATE

EX: SPEECH RECOGNITION



WHAT SHOULD YOU FOCUS ON?

NOISE  
ACCENTS  
FAR FROM MIKE

1. START QUICKLY DEV/TEST METRICS
2. GET TRAIN-SET
3. TRAIN
4. BIAS/VARIANCE ANAL
5. ERROR ANALYSIS
6. PRIORITIZE NEXT STEP

# TRAIN vs DEV/TEST MISMATCH

## AVAILABLE DATA

200 k PRO CAT PICS FROM INTERNET

10 k BLURRY CAT PICS FROM APP  
WHAT WE CARE ABT

HOW DO WE SPLIT → TRAIN/DEV/TEST?

OPTION 1: SHUFFLE ALL

205 k (TRAIN)	D	T
	1%	2.5%

PROBLEM: DEV/TEST IS NOW  
MOSTLY WEB/IMG (NOT REPR. OF END SCENARIO)

SOLUTION: LET DEV/TEST COME  
FROM APP. THEN SHUFFLE 5k  
OF APP PICS IN WEB FOR TRAIN

205 k	25	25
WEB+APP	APP	APP

## BIAS & VARIANCE W MISMATCHED TRAIN/DEV

HUMANS	~0%
TRAIN	1%
DEV ERR	10%

IS THIS DIFF  
DUE TO THE MODEL  
NOT GENERALIZING  
OR IS DEV DATA  
MUCH HARDER

A: CREATE A TRAIN-DEV SET  
THAT WE DON'T TRAIN ON

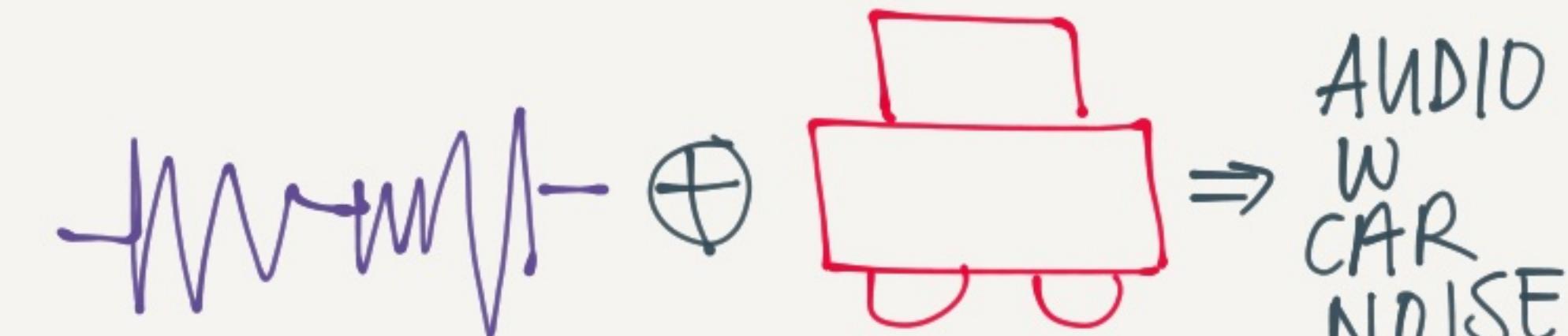
TRAIN	F	D	T
-------	---	---	---

	A	B	C	D
TRAIN	1%	1%	10%	10%
TRAIN-DEV	9%	15%	11%	11%
DEV	10%	10%	12%	20%
VARIANCE		TRAIN/DEV MISMATCH	BIAS	BIAS+DATA MISMATCH

## ADDRESSING DATA MISMATCH

EX. CAR GPS • TRAINING DATA IS 10,000H  
OF GENERAL SPEECH DATA

1. CARRY OUT MANUAL ERROR ANALYSIS  
TO UNDERSTAND THE DIFFERENCE  
(EX NOISE, STREET NUMBERS)
2. TRY TO MAKE TRAIN MORE SIMILAR  
TO DEV OR GATHER MORE DEV-LIKE  
TRAIN-DATA



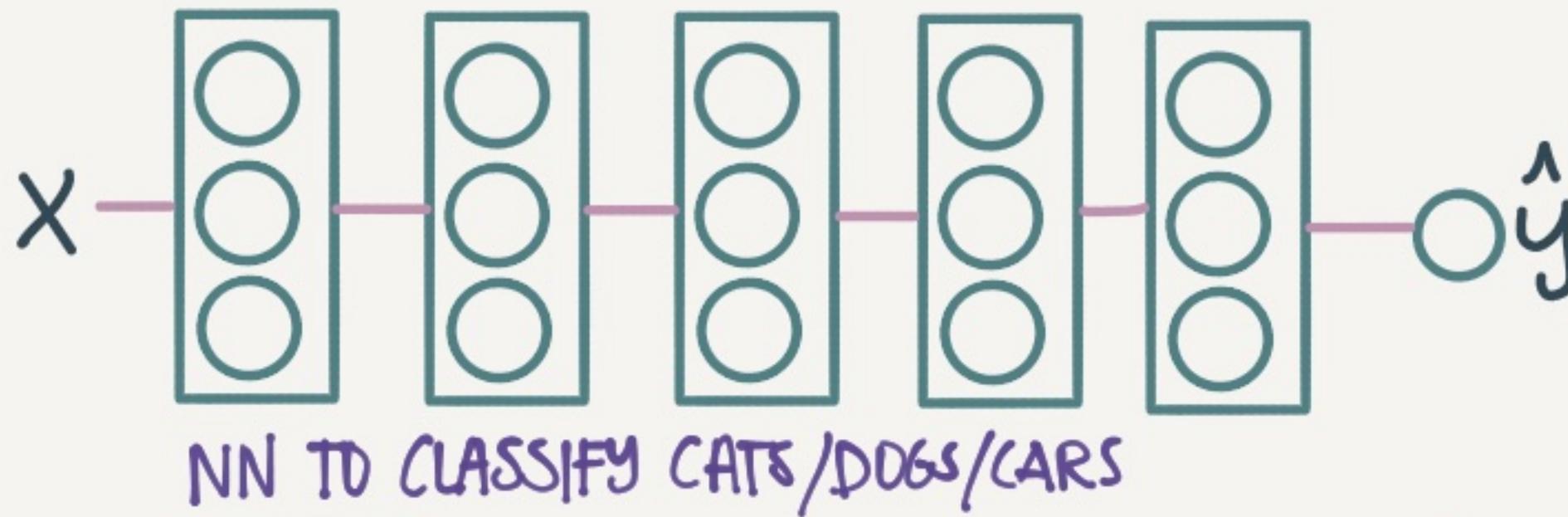
### NOTE

BE CAREFUL • IF YOU  
ONLY HAVE 1 HR OF  
CAR NOISE & APPLY IT TO 10K HR  
SPEECH YOU MAY OVERFIT TO  
THE CAR NOISE

# EXTENDED LEARNING

## TRANSFER LEARNING

PROBLEM: YOU WANT TO CLASSIFY SOME MEDICAL IMGS. YOU HAVE AN NN THAT CLASSIFIES CATS



**[OPTION 1]:** YOU ONLY HAVE A FEW RADIOLOGY IMAGES

SOLUTION: INIT W. WEIGHS FROM CAT NN  
ONLY RETRAIN LAST LAYER(S) ON RADIOLOGY IMAGES

**[OPTION 2]** YOU HAVE LOTS OF RADIOLOGY IMGS.

SOLUTION: INIT WITH WEIGHTS FROM CAT NN  
RETRAIN ALL LAYERS

THIS IS MICROSOFT CUSTOM VISION



## MULTI TASK LEARNING

TRAINING ON MULT. TASKS AT ONCE



UNLIKE SOFTMAX • MANY THINGS CAN BE TRUE

$$\text{COST: } J(w, b) = \frac{1}{m} \sum_{i=1}^m \sum_{j=1}^4 f(y_i^{(j)}, \hat{y}_i^{(j)})$$

SUMMING OVER ALL OUTP OPTIONS

WE COULD HAVE JUST TRAINED 4 NN'S INSTEAD BUT... MT LEARNING MAKES SENSE WHEN

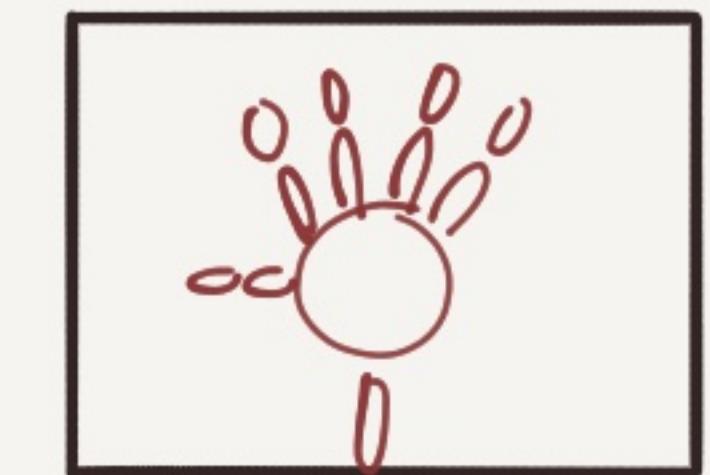
A. THE LEARNING DATA YOU HAVE FOR THE DIFF TASKS IS QUITE SIMILAR - & THE AMOUNTS (EG. 1K CARS, 1K STOP SIGNS)

B. THE SUM OF THE DATA ALLOWS YOU TO TRAIN A BIG ENOUGH NN TO DO WELL ON ALL TASKS

IN REALITY TRANSFER LEARNING IS USED MORE OFTEN

## END-TO-END LEARNING

FROM X-RAY OF CHILDS HAND TELL ME THE AGE OF THE CHILD



TYPICAL SGN:

1. LOCATE BONES TO FIND LENGTHS USING ML
2. TRAIN MODEL TO PREDICT AGE BASED ON BONE LENGTH

## END-TO-END

RADIOLOGY → CHILD AGE

PRDS:

- LET'S THE DATA SPEAK (MAYBE IT FINDS RELATIONS WE'RE UNAWARE OF)
- LESS HAND-DESIGNING OF COMPONENTS NEEDED

CONS:

- NEEDS LARGE AMTS OF ~~LABLED~~ DATA ( $X \rightarrow Y$ )
- EXCLUDES POTENTIALLY USEFUL HAND-MADE COMPONENTS

# CONVOLUTION

## FUNDAMENTALS

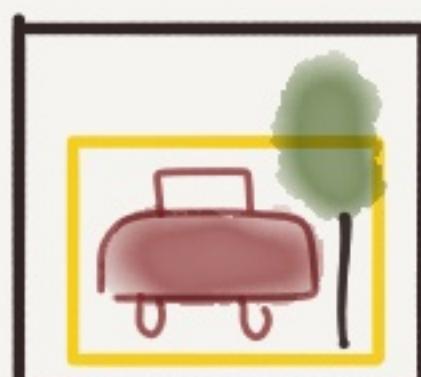
### COMPUTER VISION

IMAGE  
CLASSIFICATION



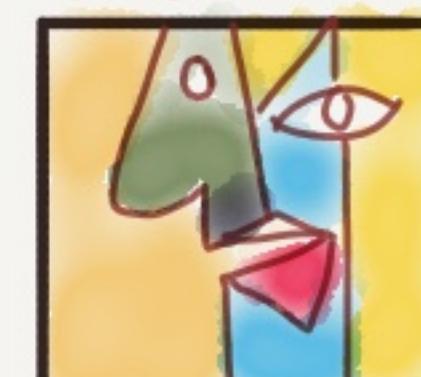
CAT OR  
NOT-CAT

OBJECT  
DETECTION



WHERE IS  
THE CAR?

NEURAL  
STYLE  
TRANSFER



PAINT ME  
LIKE PICASSO

PROBLEM: IMAGES CAN BE BIG

$$1000 \times 1000 \times 3 (\text{RGB}) = 3\text{M}$$

WITH 1000 HIDDEN UNITS WE  
NEED  $3\text{M} \times 1000 = 3\text{B}$  PARAMS

SOLUTION: USE CONVOLUTIONS

IT'S LIKE SCANNING OVER YOUR  
IMG WITH A MAGNIFYING GLASS  
OR FILTER



ALSO SOLVES THE PROBLEM  
THAT THE CAT IS NOT  
ALWAYS IN THE SAME  
LOCATION IN THE IMB

## CONVOLUTION

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

INPUT 6x6 IMAGE

$$\begin{array}{c} 3+1+2+0+0+0-1-8-2=-5 \\ (3 \times 1) \\ \downarrow \\ \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline \end{array} \\ \text{FILTER } 3 \times 3 \\ \downarrow \\ \text{CONVOLUTION} \end{array} = \begin{array}{|c|c|c|} \hline -5 & 4 & 0 & 8 \\ \hline -16 & -2 & 2 & 3 \\ \hline 0 & -2 & -4 & -7 \\ \hline -3 & -2 & -3 & -16 \\ \hline \end{array} \text{ OUTPUT } 4 \times 4 \text{ IMAGE}$$

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

INPUT 6x6 IMAGE

$$\begin{array}{c} \text{VERTICAL} \\ \text{EDGE DETECTOR} \\ \downarrow \\ \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline \end{array} \\ \text{FILTER } 3 \times 3 \\ \downarrow \\ \text{DETECTED} \\ \text{EDGE IN THE MIDDLE} \end{array} = \begin{array}{|c|c|c|} \hline 0 & 30 & 30 & 0 \\ \hline 0 & 30 & 30 & 0 \\ \hline 0 & 30 & 30 & 0 \\ \hline 0 & 30 & 30 & 0 \\ \hline \end{array} \text{ OUTPUT } 4 \times 4 \text{ IMAGE}$$

THIS IS LIKE ADDING  
AN 'INSTA' FILTER THAT  
JUST SHOWS OUTLINES

WE COULD HARD-CODE FILTERS · JUST LIKE WE  
CAN HARD-CODE HEURISTIC RULES ... BUT... A MUCH BETTER  
WAY IS TO TREAT THE FILTER# AS PARAMS  
TO BE LEARNED

$w_1$	$w_2$	$w_3$
$w_4$	$w_5$	$w_6$
$w_7$	$w_8$	$w_9$

# CONVENTIONAL NEURAL NETS · COURSEPART

## PADDING

PROBLEM: IMAGES SHRINK  
 $6 \times 6 \rightarrow 3 \times 3 \rightarrow 4 \times 4$

PROBLEM: EDGES GET LESS 'LOVE'

SOLUTION: PAD W. A BORDER OF 0s BEFORE CONVOLVING

0	0	0	6	0	0	6	0
0	3	0	1	2	7	4	0
0	1	5	8	9	3	1	0
0	2	7	2	5	1	3	0
0	0	1	3	1	7	8	0
0	4	2	1	6	2	8	0
0	2	4	5	2	3	9	0
0	0	0	0	0	0	0	0

TWO COMMONLY USED  
PADDING OPTIONS

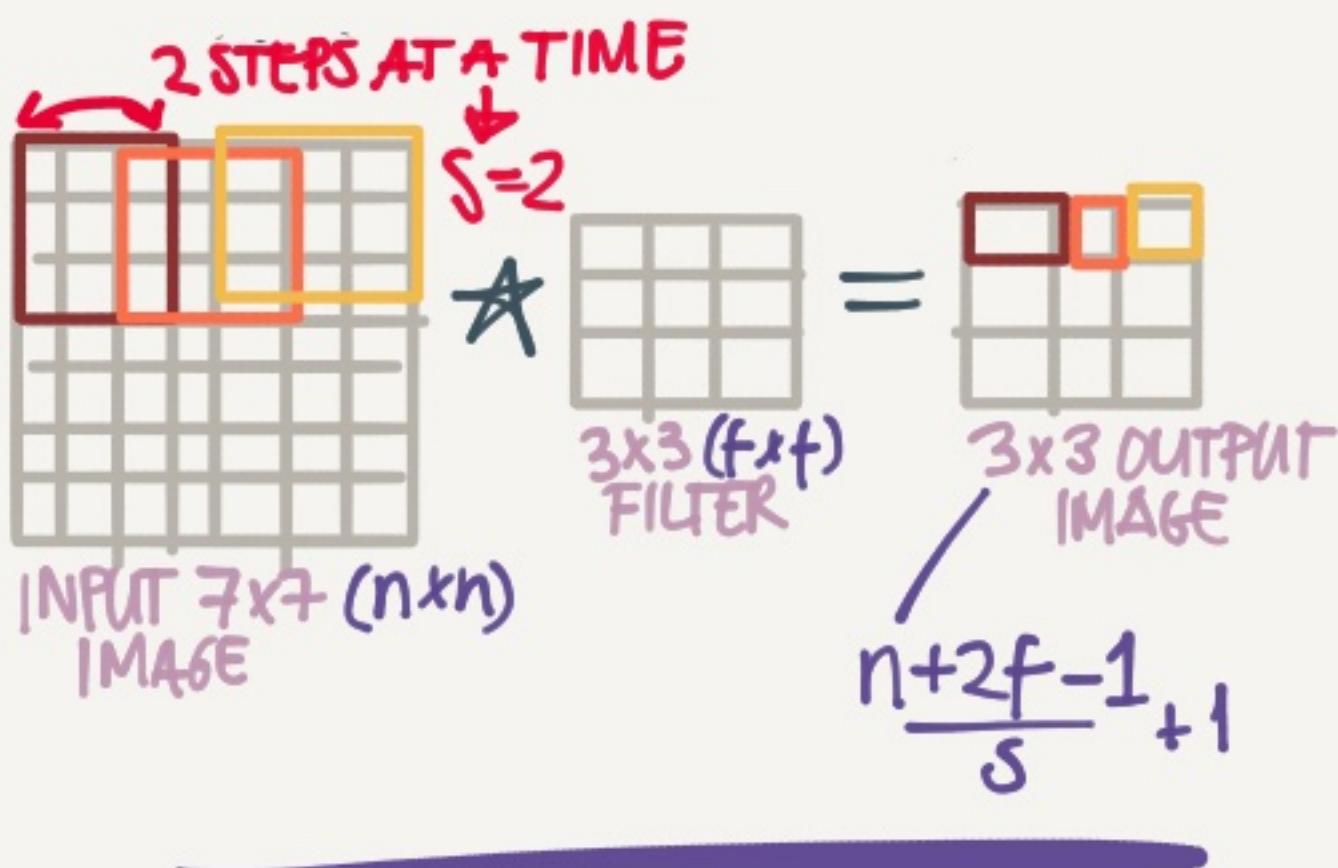
(HOW MUCH TO PAD)

$$\begin{aligned} \text{'VALID'} &\Rightarrow P=0 & \text{NO PADDING} \\ \text{'SAME'} &\Rightarrow P=\frac{f-1}{2} & \text{OUTPUT SIZE = INPUT SIZE} \\ && \text{FILTER SIZE} \end{aligned}$$

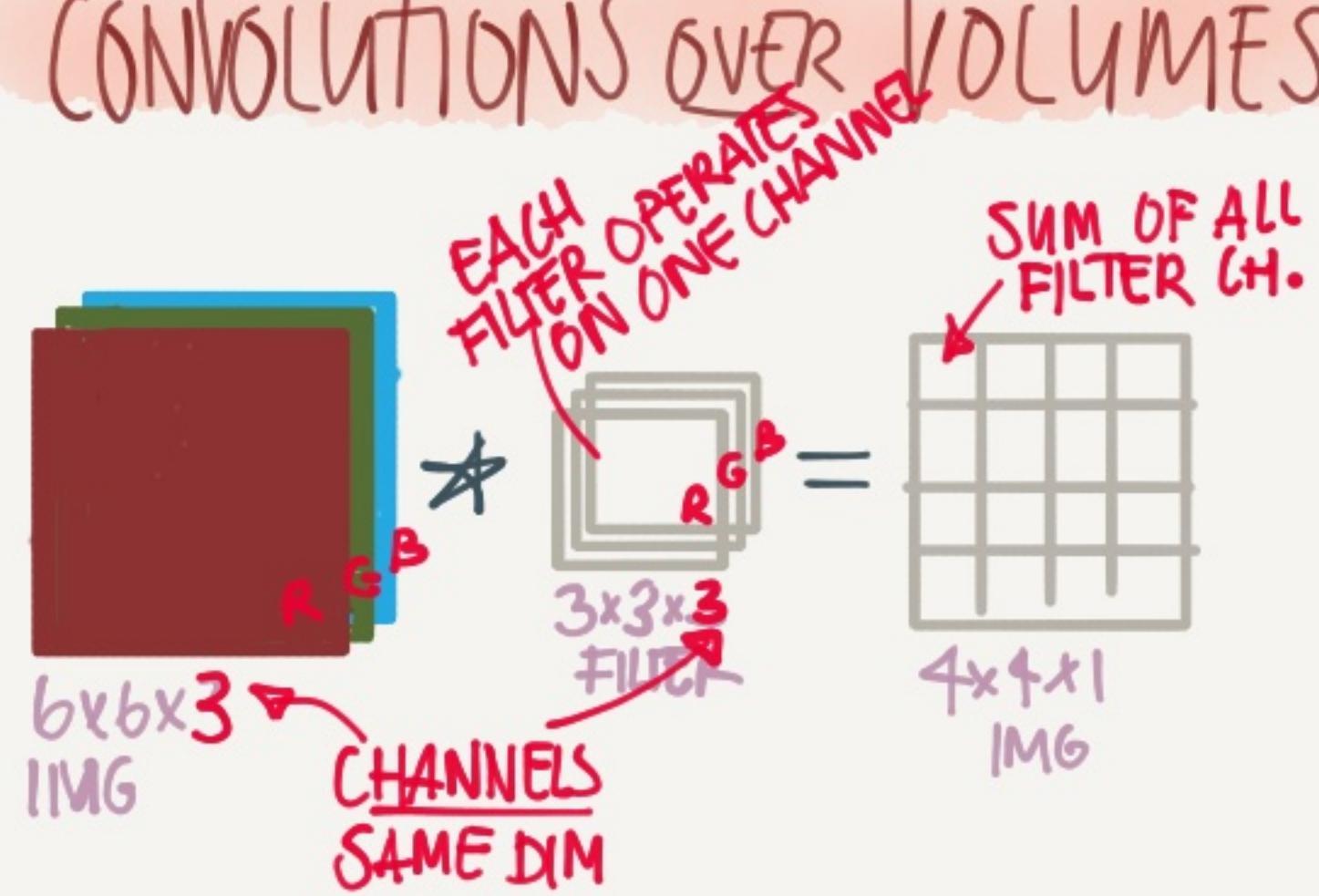
**NOTE** ALL CONVOLUTION IDEAS CAN BE  
APPLIED TO 1D AS WELL LIKE  
EKG SIGNALS · AND 3D LIKE CT·SCANS

## STRIDE

WHAT PACE YOU SCAN WITH



CONVOLUTIONS OVER VOLUMES

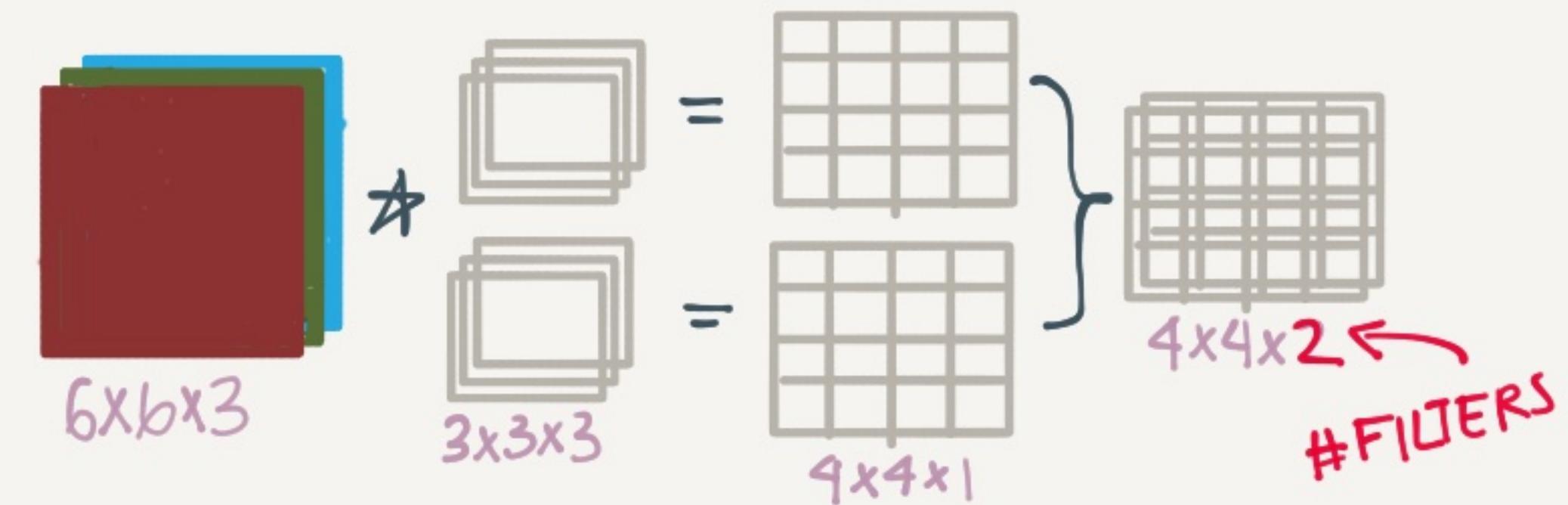


THIS ALLOWS US TO DETECT FEATURES  
IN COLOR IMAGES FOR EXAMPLE

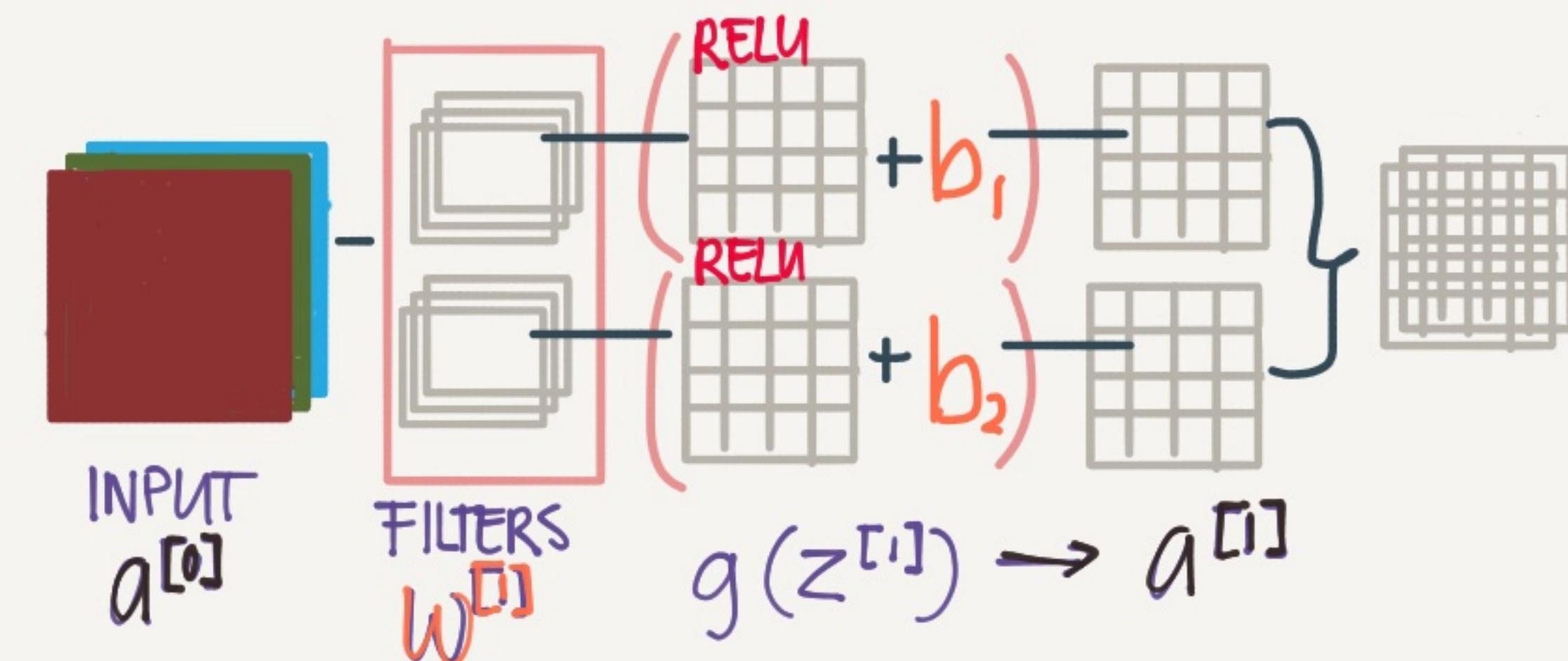
MAYBE WE WANT TO FIND ALL  
EDGES OR MAYBE ORANGE BLOBS

## MULTIPLE FILTERS

DETECTING MULTIPLE FEATURES AT A TIME



## ONE CONV. NET LAYER



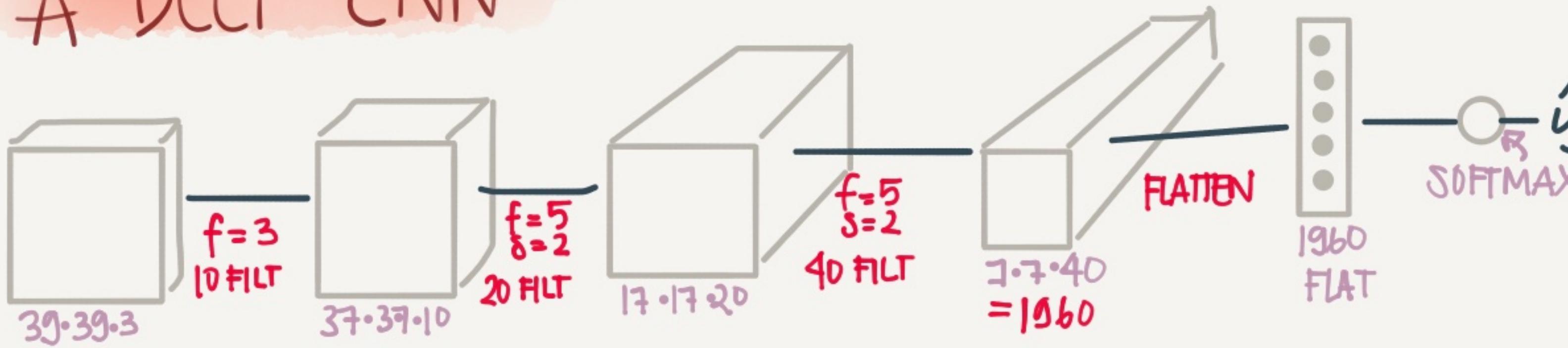
**NOTE** IT DOESN'T MATTER HOW BIG THE  
INPUT IS - THE LEARNABLE PARAMS  $W$  &  $b$   
ONLY DEPEND ON THE # OF FILTERS  
AND THEIR SIZES.

$$W = 3 \cdot 3 \cdot 3 \cdot 2 = 54 \quad \left\{ \begin{array}{l} 56 \text{ PARAMS} \\ \text{TO LEARN} \end{array} \right.$$

$$b = 2$$

© TessFerrandez

# A DEEP CNN



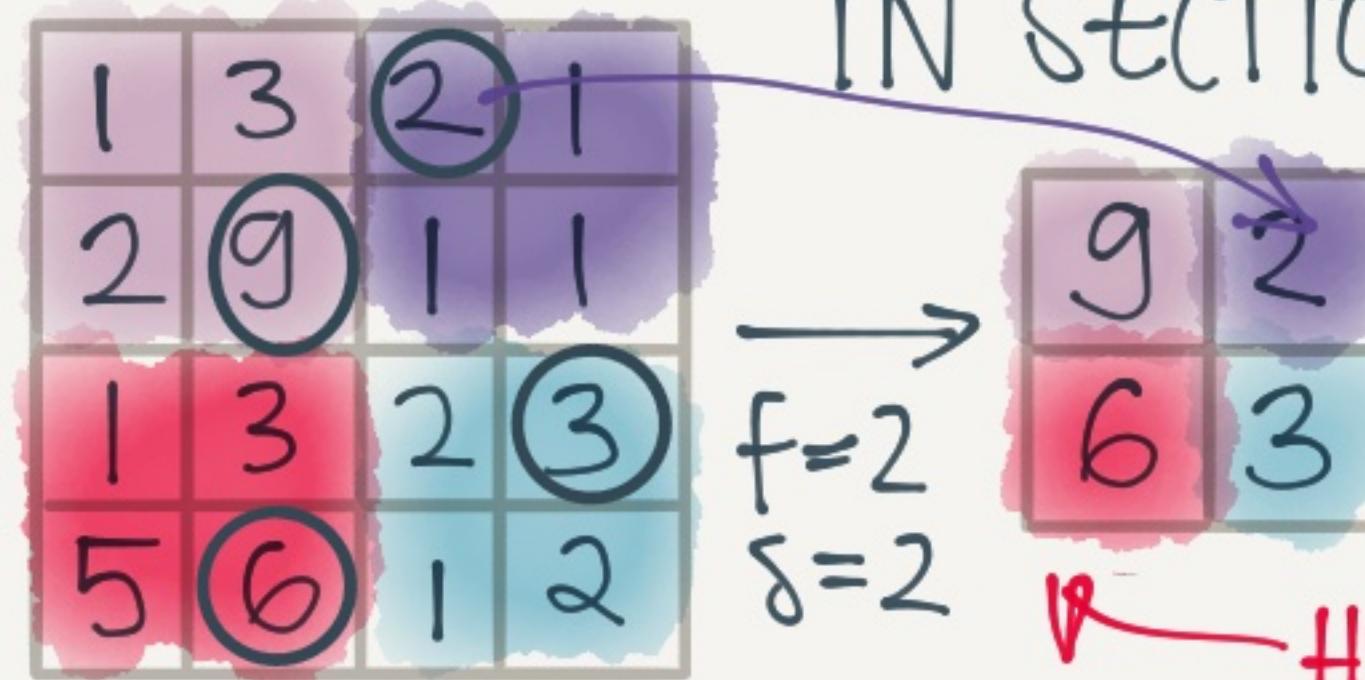
A LOT OF THE WORK IS FIGURING OUT HYPERPARAMS  
 $= \# \text{FILTERS}, \text{STRIDE}, \text{PADDING} \text{ ETC}$

TYPICALLY  $\text{SIZE} \rightarrow \text{TREND DOWN}$   
 $\# \text{FILTERS} \rightarrow \text{TREND UP}$

## TYPICAL CONV.NET LAYERS

CONVOLUTION  
POOLING  
FULLY CONNECTED

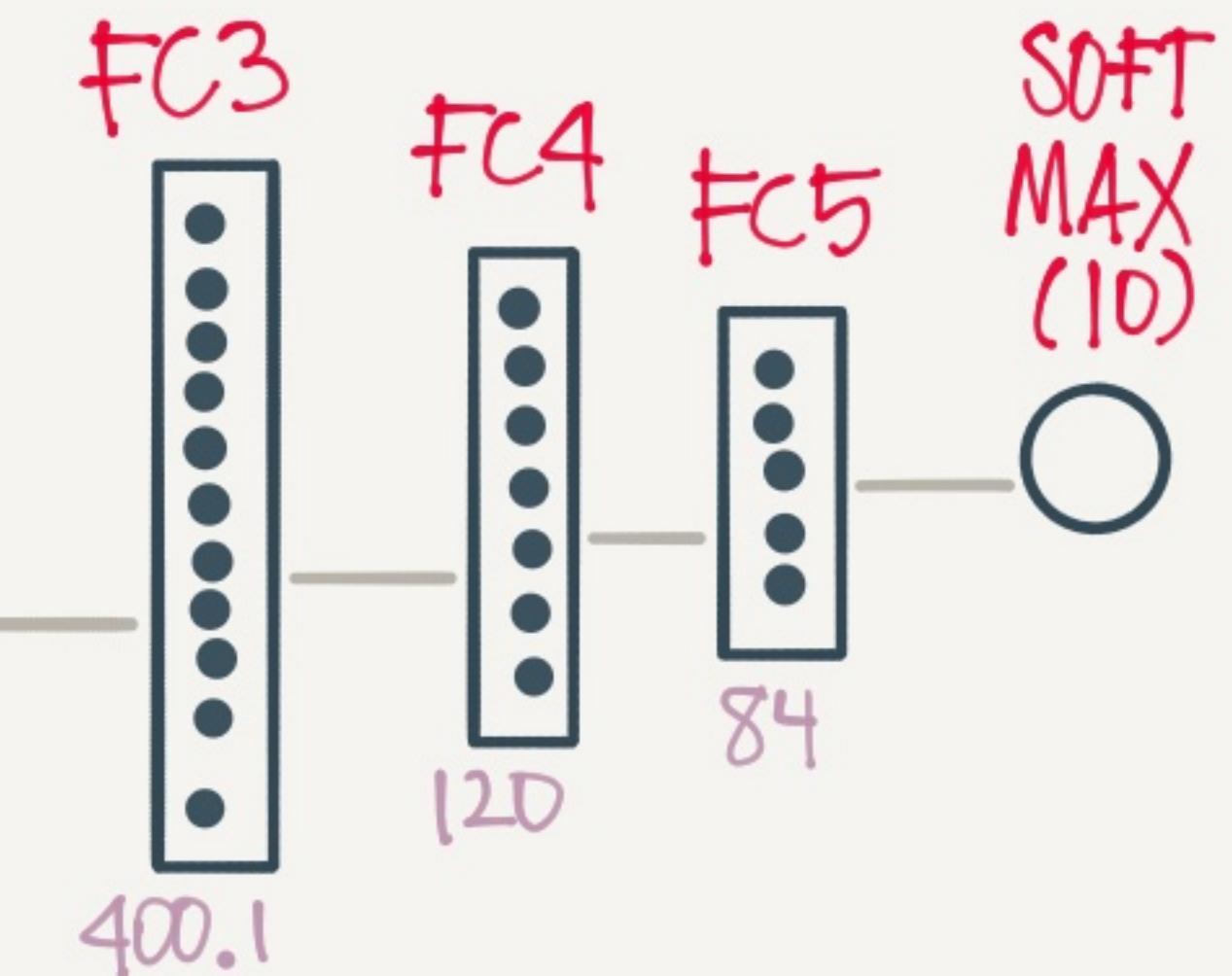
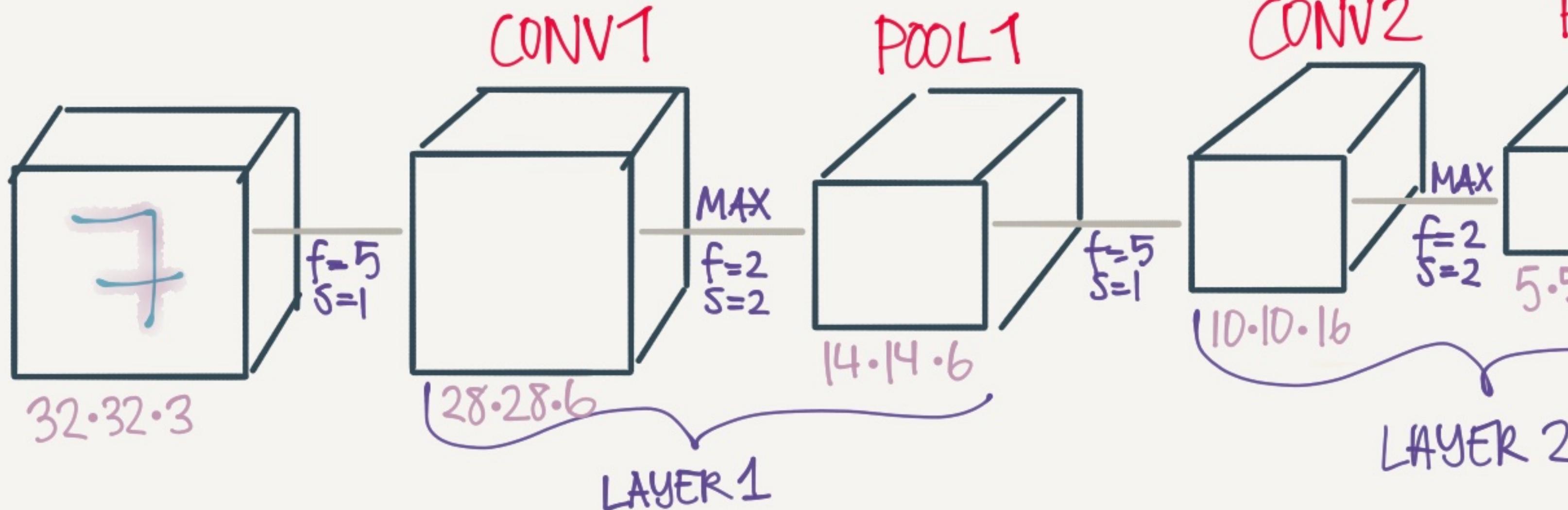
POOLING (MAX)  
FIND MAX VAL IN SECTION



- \* REDUCES SIZE OF REPRES.
- \* SPEEDS UP COMPUTATION
- \* MAKES SOME OF THE DETECTED FEAT. MORE ROBUST

## CONV NET EXAMPLE BASED ON LeNet-5

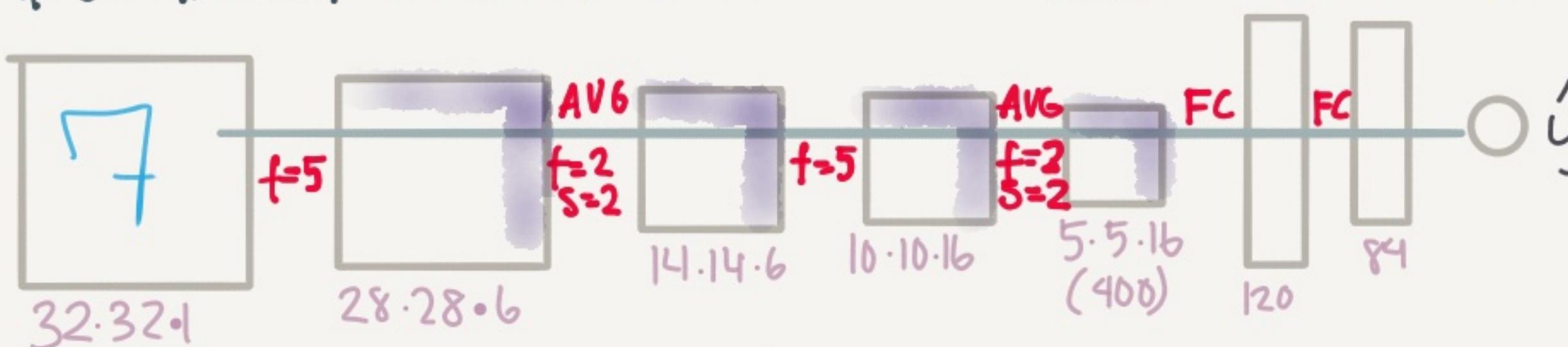
DETECTING HANDWRITTEN DIGITS



# CLASSIC CONV. NETS

## LeNet-5

DOCUMENT CLASSIFICATION



$\approx 60k$  PARAMETERS

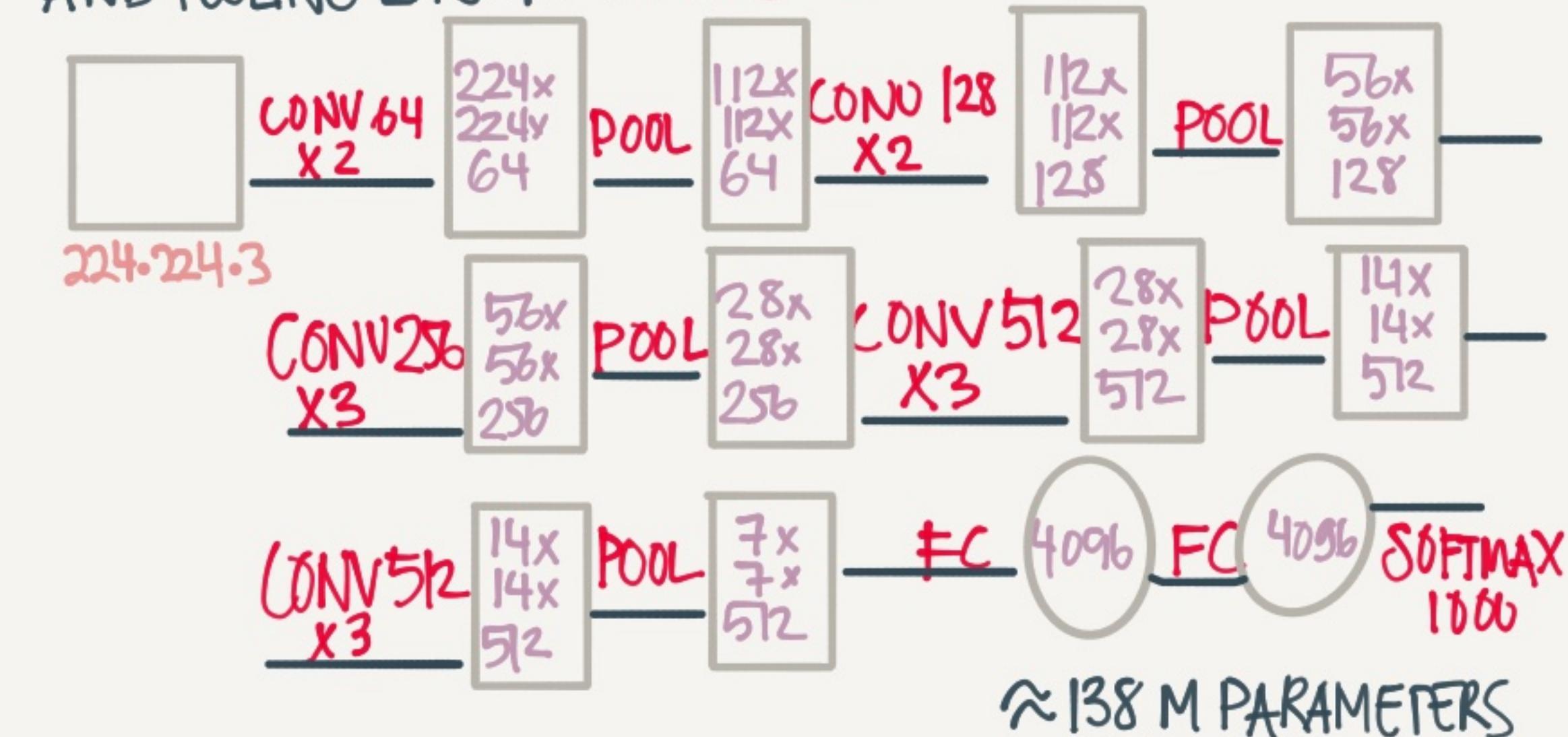
TRENDS: HEIGHT/WIDTH GO DOWN  
CHANNELS GO UP

COMMON PATTERN: A COUPLE OF CONV(1<sup>st</sup>)/POOL LAYERS FOLLOWED BY A FEW FC

OLD STUFF: USED AVG POOLING INST. OF MAX  
PADDING WAS NOT VERY COMMON  
IT USED SIGMOID/TANH INST OF RELU

## VGG-16

ALL CONV. LAYERS HAVE SAME PARAMS  
 $f=3 \times 3$   $s=1$   $p=\text{SAME}$   
AND POOLING LAYER  $2 \times 2$   $s=2$



$\approx 138$  M PARAMETERS

- VERY DEEP
- EASY ARCHITECTURE
- # FILTERS DOUBLE 64, 128, 256, 512

## AlexNet

IMAGE CLASSIFICATION



$\approx 60$  M PARAMETERS

- SIMILAR TO LeNet BUT MUCH BIGGER
- USES RELU
- THE NN THAT GOT RESEARCHERS INTERESTED IN VISION AGAIN

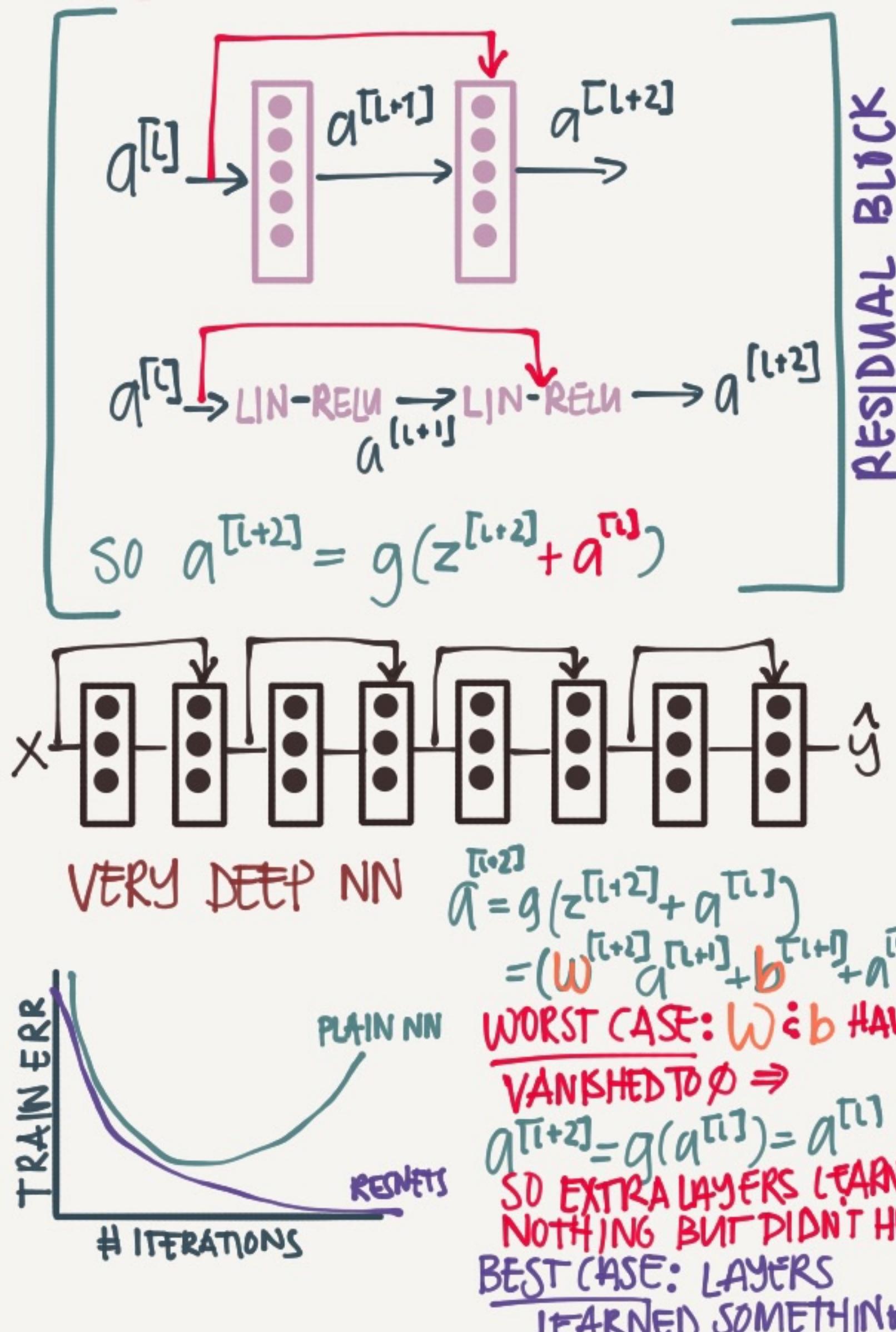
# CONVENTIONAL NEURAL NETS · COURSE

# SPECIAL NETWORKS

## ResNets

PROBLEM: DEEP NN OFTEN SUFFER PROBLEMS W VANISHING OR EXPLODING GRADIENTS

SOLUTION: RESIDUAL NETS



## NETWORK IN NETWORK (1x1 CONVOLUTION)

$$\begin{array}{rrrr} 6 & 5 & 3 & 2 \\ 4 & 1 & 9 & 5 \\ 5 & 8 & 2 & 4 \\ 0 & 3 & 6 & 1 \end{array} \star \boxed{2} = \begin{array}{rrrr} 12 & 10 & 6 & 4 \\ 8 & 2 & 18 & 10 \\ 10 & 16 & 4 & 8 \\ 0 & 6 & 12 & 2 \end{array}$$

### 1x1 CONVOLUTION

IT SEEMS PRETTY USELESS, BUT IT ACTUALLY SERVES 2 PURPOSES

### 1. NETWORK IN A NETWORK



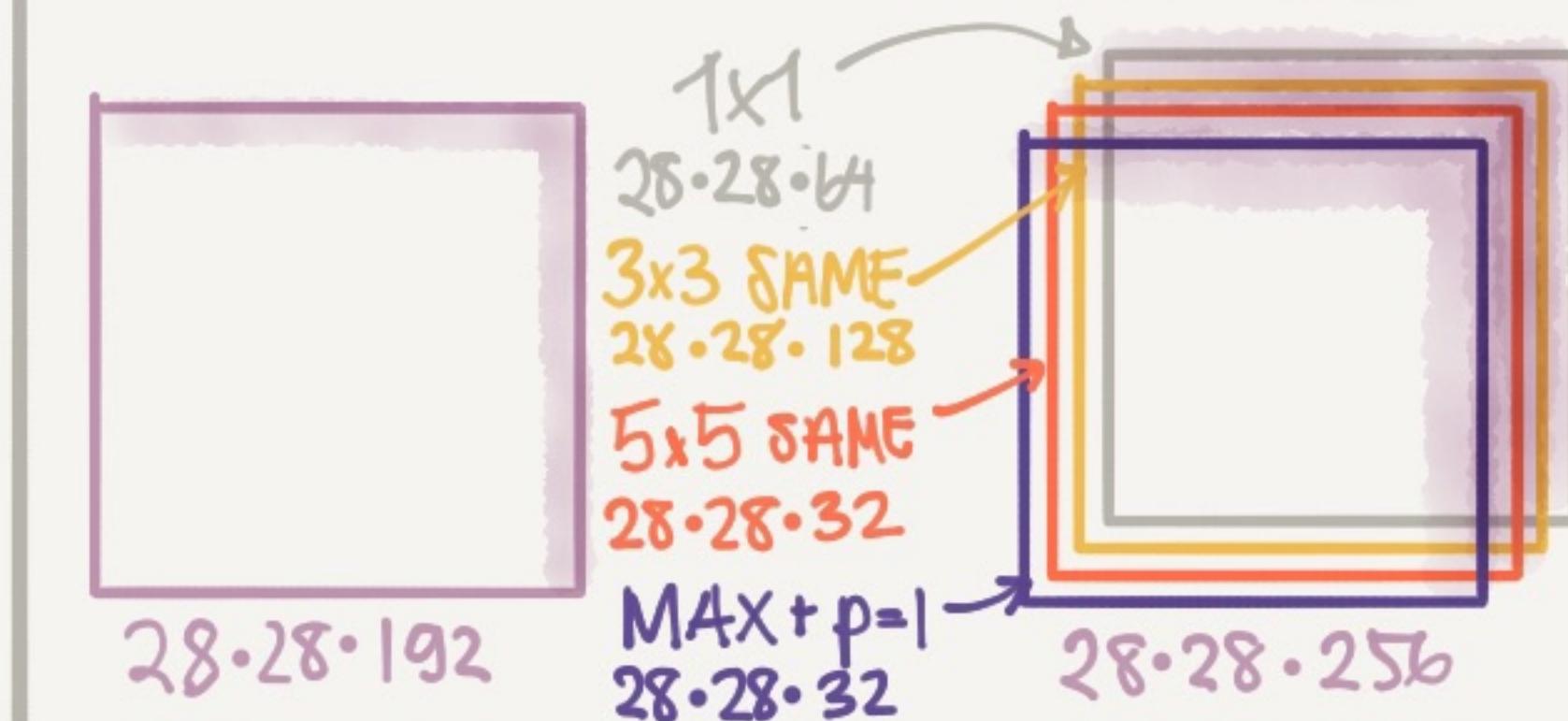
LEARNS COMPLEX, NON-LINEAR RELATIONSHIPS ABOUT A SLICE OF A VOLUME

### 2. REDUCING # CHANNELS

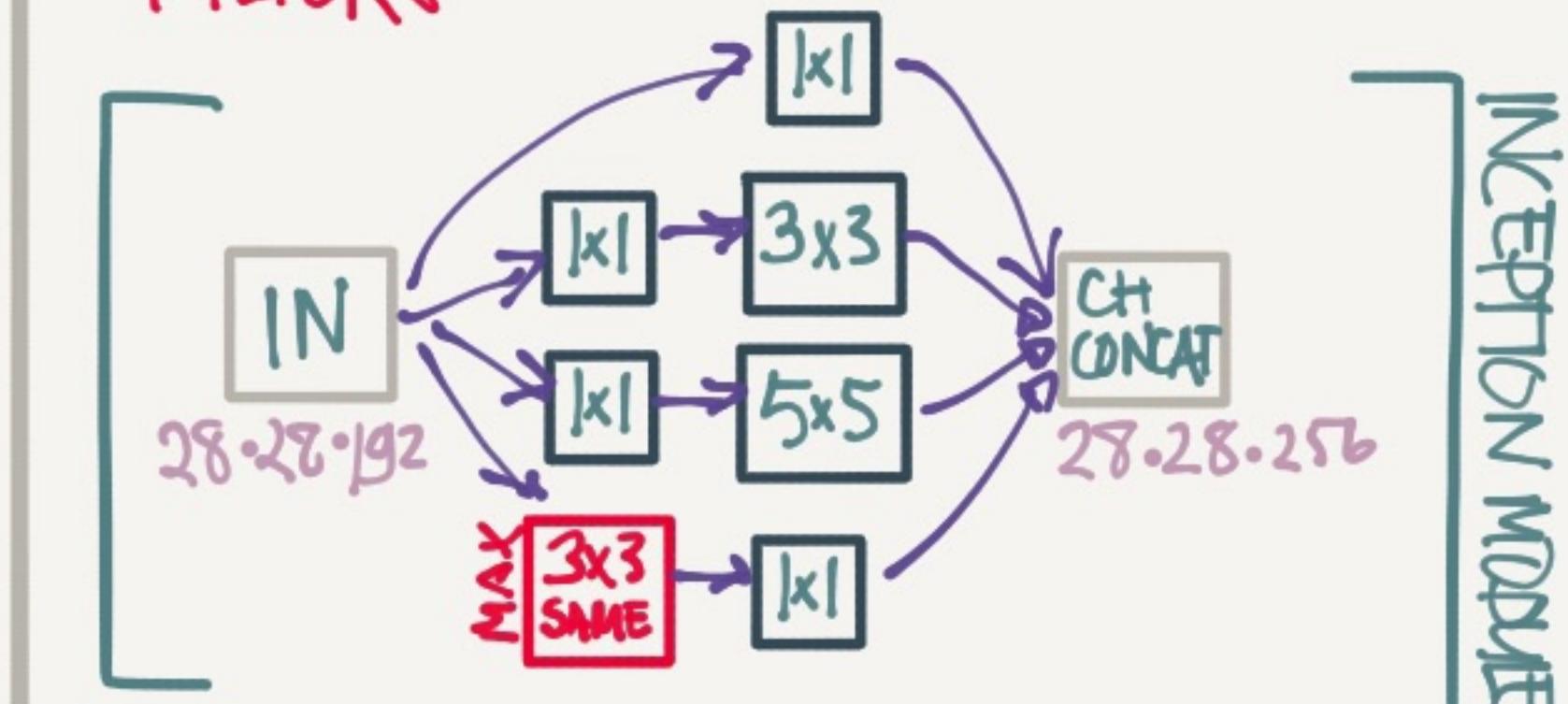
$$\begin{array}{r} 28 \cdot 28 \cdot 192 \\ \star \boxed{1 \times 1 \times 92} \\ 32 \text{ FILT} \end{array} = \begin{array}{r} 28 \cdot 28 \cdot 32 \end{array}$$

## INCEPTION NETWORKS

INSTEAD OF CHOOSING A  $1 \times 1, 3 \times 3, 5 \times 5$  OR A POOLING LAYER - CHOOSE ALL



PROBLEM: VERY EXPENSIVE TO COMPUTE  
SOLUTION: SHRINK THE # CHANNELS W A  $1 \times 1$  CONV BEFORE APPLYING ALL THE FILTERS



TO BUILD AN INCEPTION NETWORK YOU MAINLY STACK A BUNCH OF INCEPTION MODULES



INCEPTION THE MOVIE

© TessFerrandez

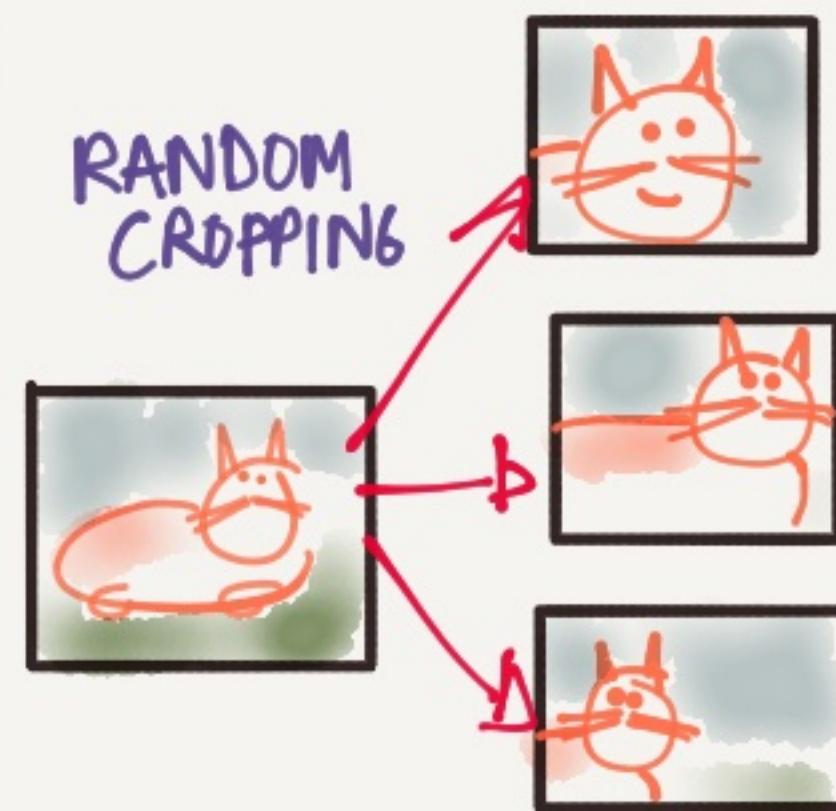
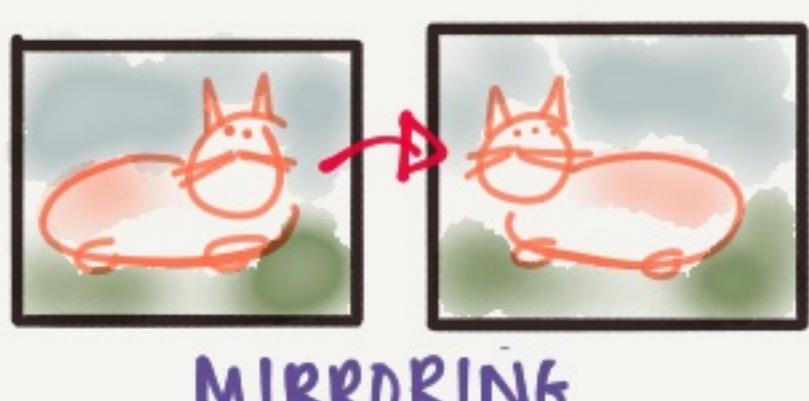
# PRACTICAL ADVICE

## USE OPEN SOURCE IMPLEMENTATIONS

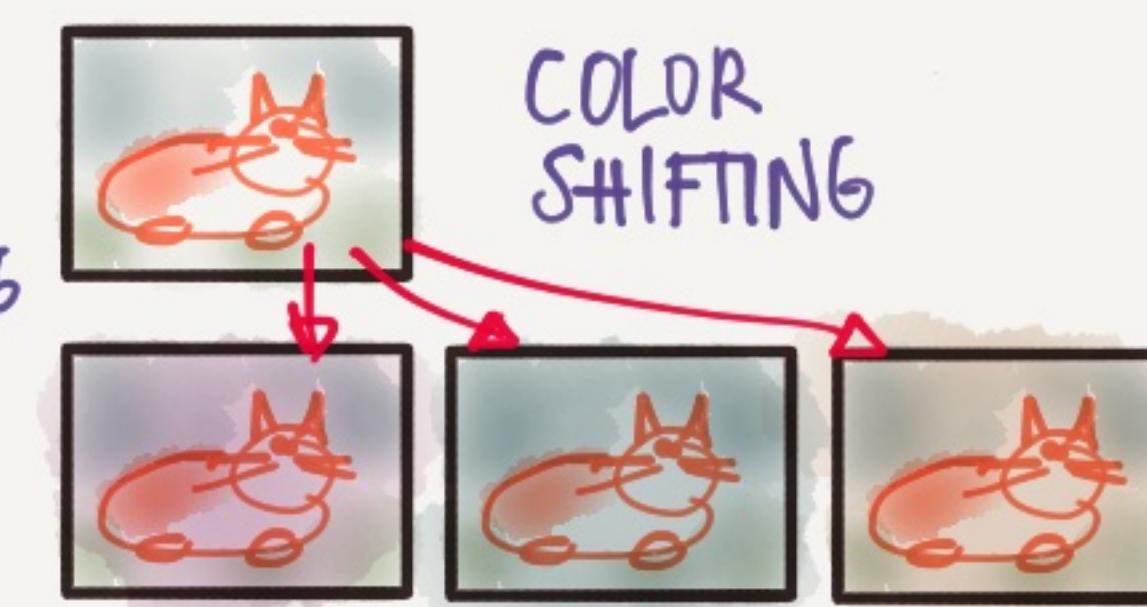
SOME OF THE PAPERS ARE HARD TO IMPLEMENT FROM SCRATCH - USING OS YOU CAN REUSE OTHER PPLS WORK  
DON'T FORGET TO CONTRIBUTE

## DATA AUGMENTATION

WE ALMOST ALWAYS NEED MORE DATA TO TRAIN ON



RANDOM CROPPING  
ROTATION  
SHEARING  
LOCAL WARPING  
...

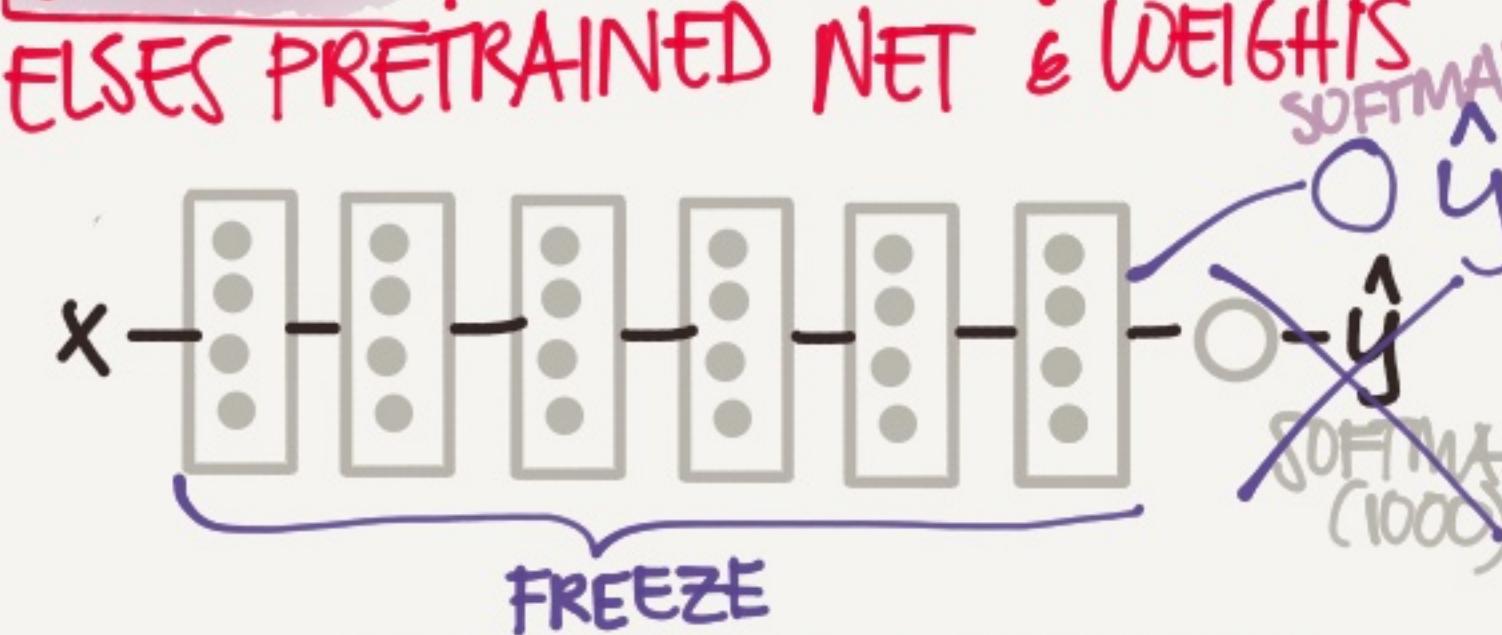


## TRANSFER LEARNING



WANT TO TRAIN A CLASSIFIER FOR YOUR CATS BUT DON'T HAVE ENOUGH PICTURES

**SOLUTION** DOWNLOAD SOMEONE ELSE'S PRETRAINED NET & WEIGHTS



FREEZE THE PARAMS, AND JUST REPLACE THE SOFTMAX LAYER WITH YOUR OWN & TRAIN

IF YOU HAVE MORE PICS • RETRAIN A FEW OF THE LATER LAYERS (MAYBE INITIALIZING WITH THE PRETRAINED WEIGHTS)

## STATE OF COMPUTER VISION

WE HAVE LOTS OF DATA

- SPEECH RECDG.

- IMAGE RECOGNITION

- OBJECT DETECTION  
IMGS IN LABELED BOXES

WE HAVE LITTLE LABELED DATA

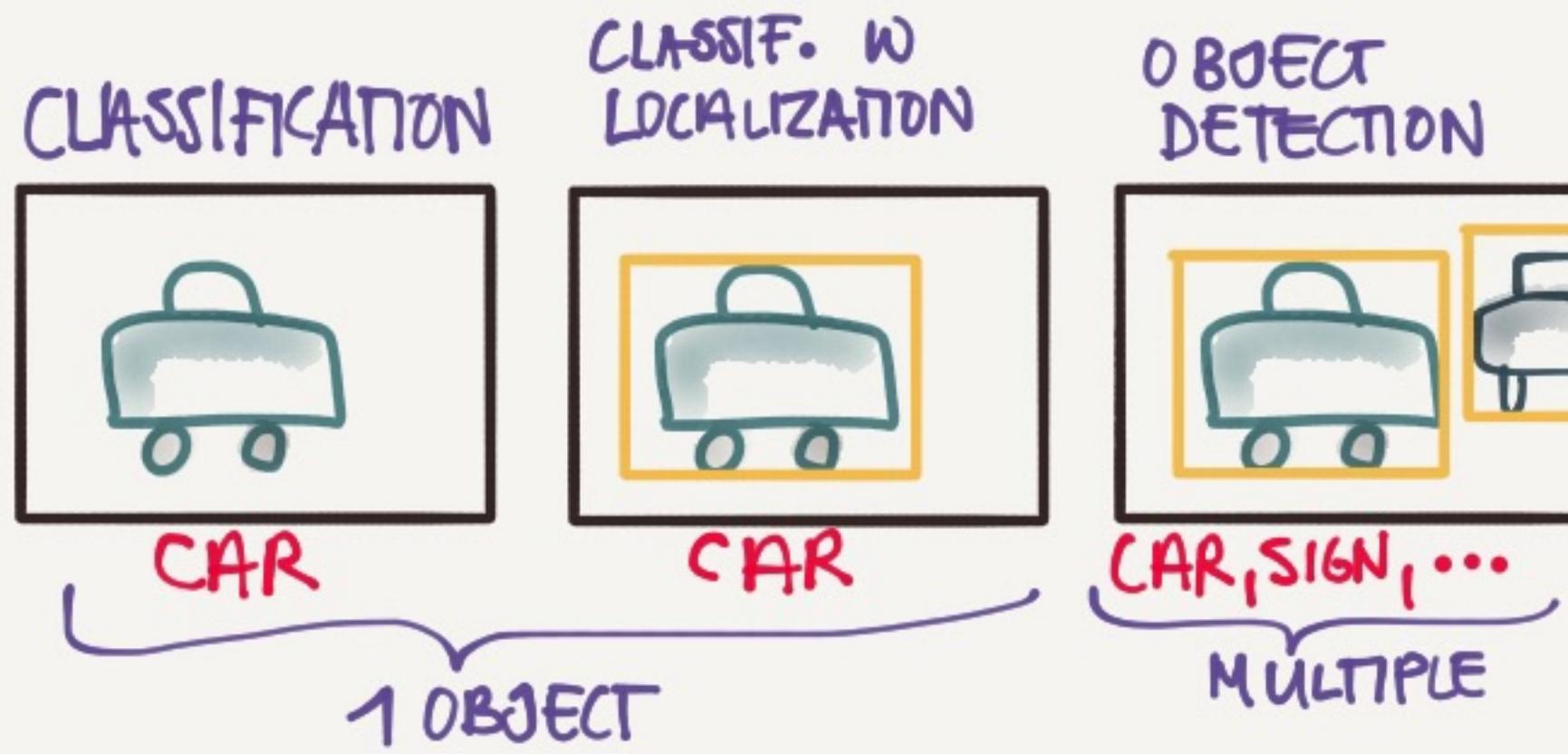
MORE HAND ENGINEERING

## TIPS FOR DOING WELL ON BENCHMARKS/COMPETITIONS

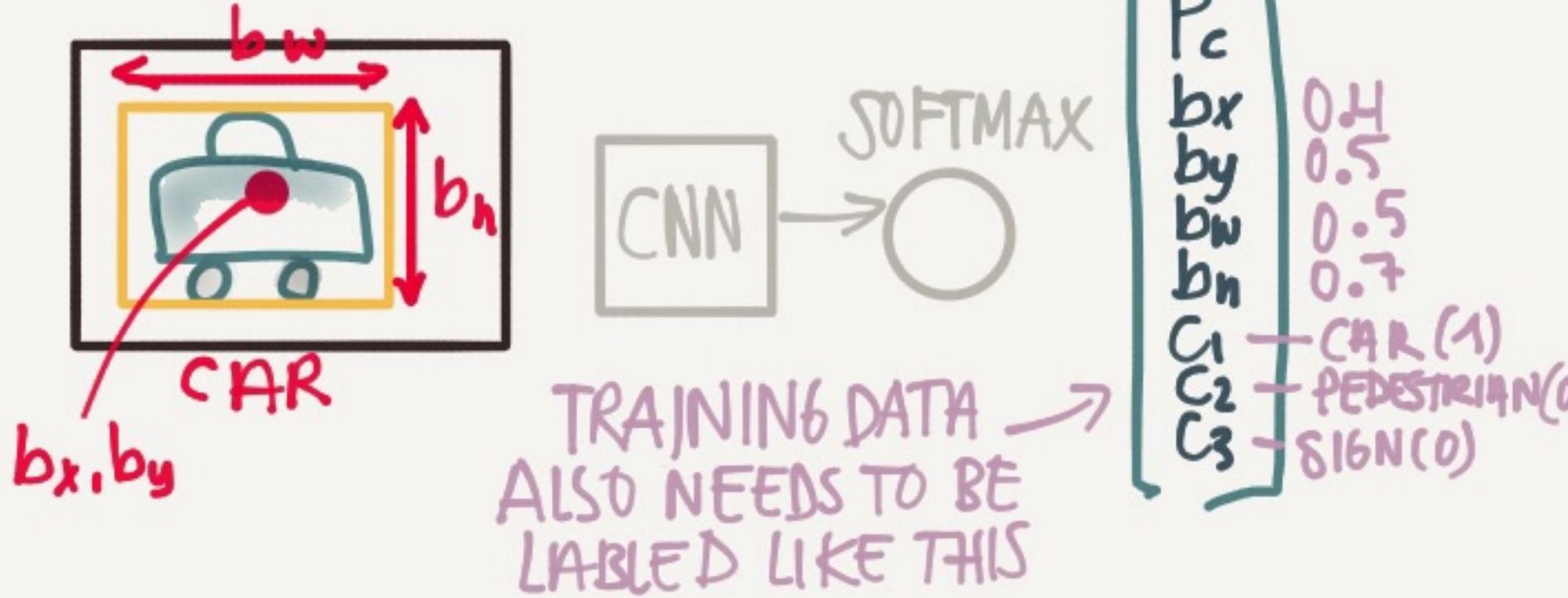
- \*ENSEMBLING.  
AVG OUTPUTS FROM MULT NN
- \*MULTI-CROP AT TEST TIME  
AVG OUTPUTS FROM MULTIPLE CROPS OF THE IMAGE

IN PRACTICE THEY ARE NOT USED IN PRODUCTION BECAUSE THEY ARE COMPUTE & MEM EXPENSIVE

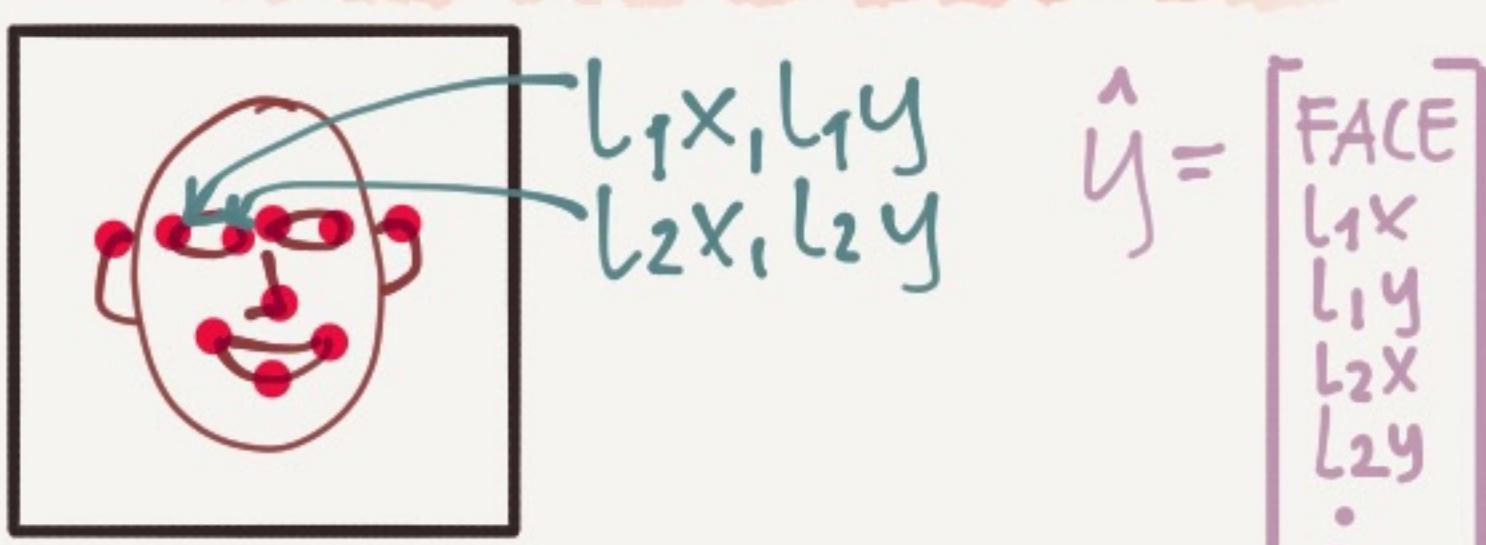
# DETECTION ALGORITHMS



## OBJECT LOCALIZATION



## LANDMARK DETECTION



TO DETECT LANDMARKS IN THE FACE (CORNER OF MOUTH ETZ) LABEL THE X, Y COORDS OF THE LANDMARK

USED FOR SENTIMENT ANALYSIS & FOR EFFECTS LIKE PLACING CROWN ON HEAD ETZ.

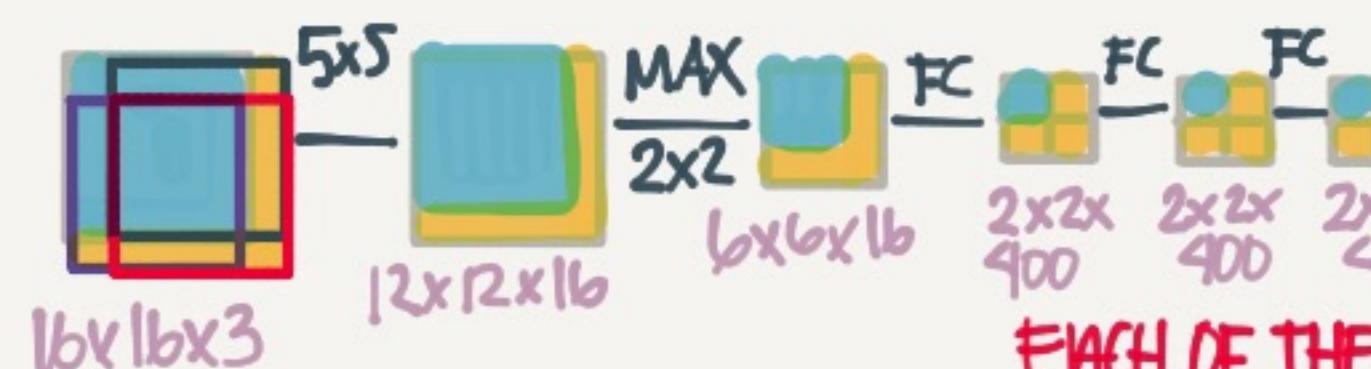
## SLIDING WINDOWS DETECTION



1. CREATE SLIGHTLY CROPPED IMGS OF CARS (LOTS)
2. SLIDE A WINDOW OVER THE IMG. & CLASSIFY THIS WINDOW CAR(1/0) AGAINST YOUR OTHER CARS
3. REPEAT WITH SLIGHTLY LARGER WINDOW SIZE

PROBLEM: VERY EXPENSIVE (TO COMPUTE)

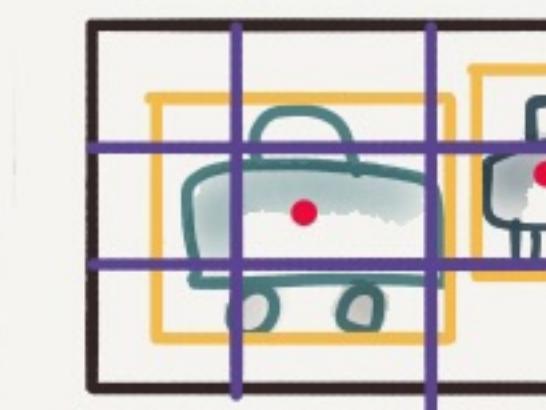
SINCE ADJ WINDOWS SHARE A LOT OF THE COMPUTATIONS WE CAN DO THIS MUCH CHEAPER W CONVOLUTIONS



NOW WE JUST PASS THROUGH ONCE AND CALC ALL AT THE SAME TIME

EACH OF THE 4 VALS ARE RESULTS FOR EACH OF THE 4 WINDOWS

## YOLO: You Only Look Once



1. SPLIT IMG INTO  $X(9)$  GRID CELLS
  2. FOR EACH CELL, SAY IF IT CONTAINS CAR + BOUNDING BOX (IF CELL CONTAINS THE MD POINT)
- $X \times 9 = 3 \times 3 \times 8$

HOW DO YOU KNOW HOW GOOD IT IS?

HOW GOOD IS THE RED SQUARE?



$$IOU = \frac{\text{SIZE OF INTERSECTION}}{\text{SIZE OF UNION}}$$

INTERSECTION OVER UNION

GENERALLY • IF  $IOU \geq 0.5$  IT IS REGARDED AS CORRECT

WHAT IF MULTIPLE SQUARES CLAIM THE SAME CAR?

## NON-MAX SUPPRESSION

IF TWO BOUNDING BOXES HAVE A HIGH IOU - PICK THE ONE W HIGHEST  $P_c$  - GET RID OF THE REST.



## ANCHOR BOXES

ANCHOR BOXES LET YOU ENCODE MULTIPLE OBJECTS IN THE SAME SQUARE



© TessFerrandez

# CONVENTIONAL NEURAL NETS · COURSE

# FACE RECOGNITION

FACE  
VERIFICATION



IS THIS PETE?  
99% ACC  $\Rightarrow$   
PRETTY GOOD

FACE·  
RECOGNITION



WHO IS THIS?  
(OUT OF K PERSONS)  
IF K = 100 NEED  
MUCH HIGHER THAN  
99%

## ONE-SHOT LEARNING

NEED TO BE ABLE TO RECOGNIZE  
A PERSON EVEN THOUGH YOU ONLY  
HAVE ONE SAMPLE IN YOUR DB.

YOU CAN'T TRAIN A CNN WITH  
A SOFTMAX (EACH PERSON) BECAUSE

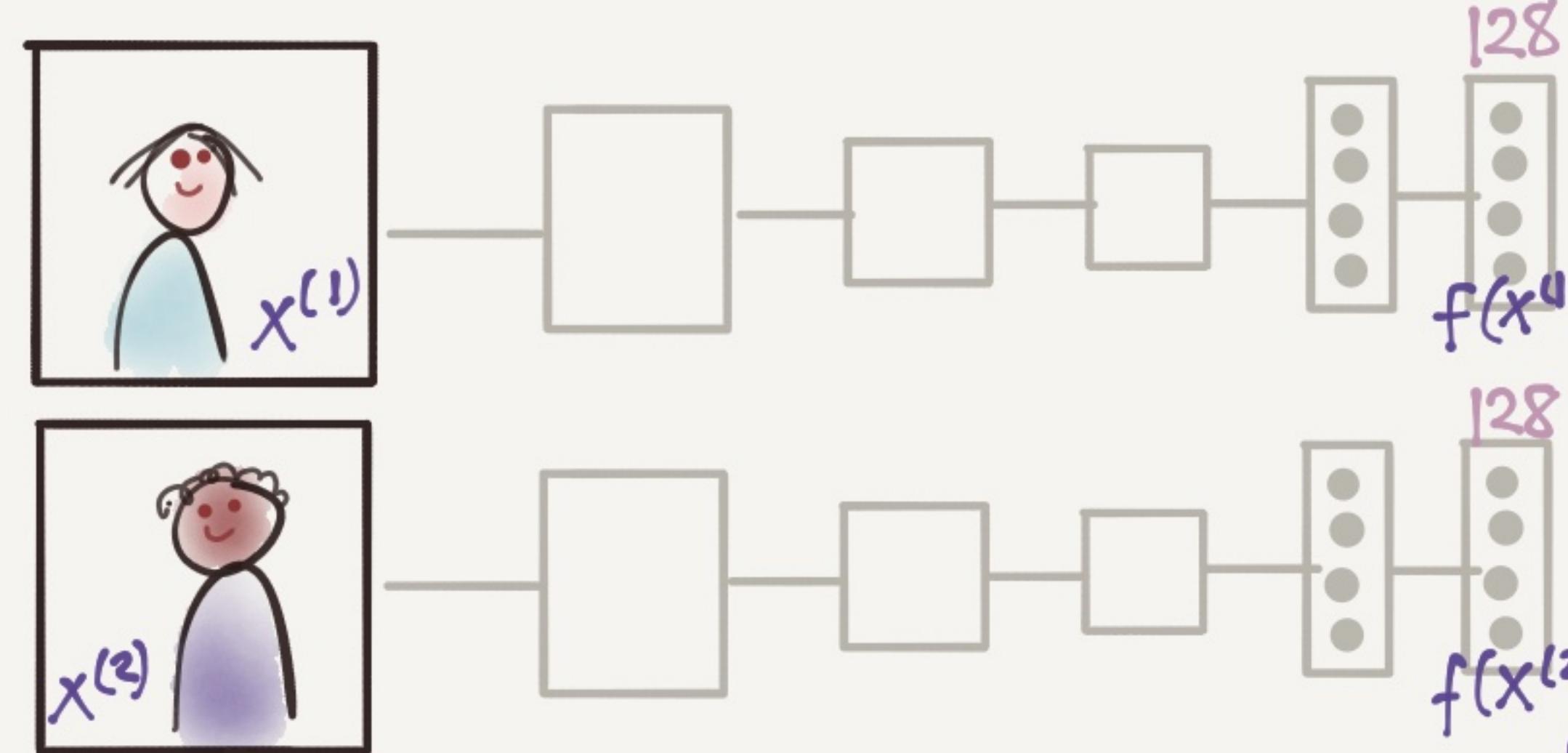
- (A) YOU DON'T HAVE ENOUGH SAMPLES
- (B) IF A NEW PERSON JOINS YOU  
NEED TO RETRAIN THE NETWORK

**SOLUTION**: LEARN A SIMILARITY  
FUNCTION

$$d(\text{img}1, \text{img}2) = \text{degree of difference}$$

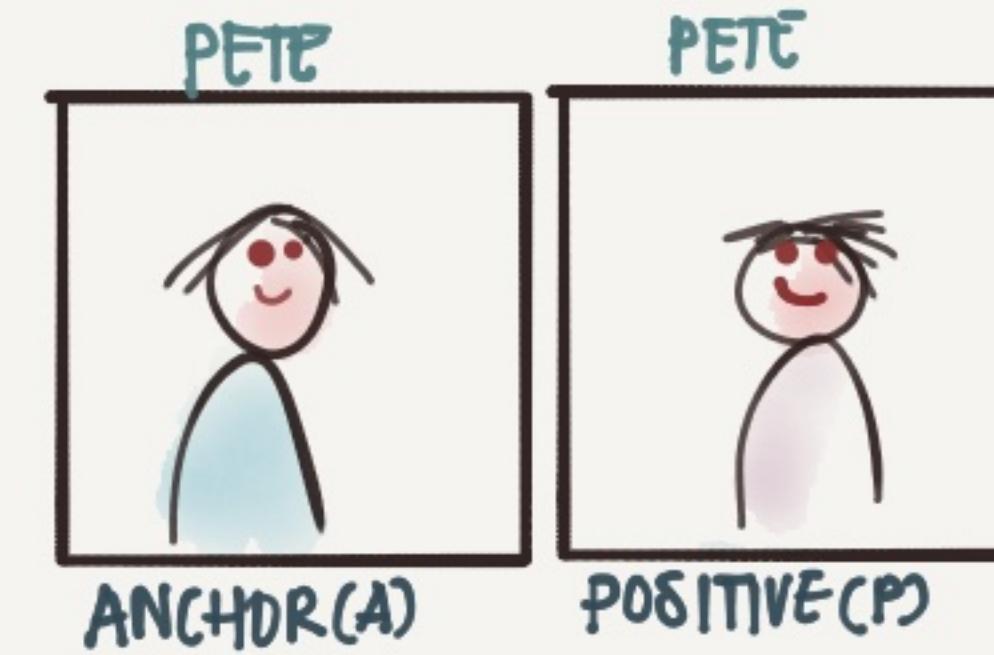
BUT HOW DO YOU LEARN THIS?

## SIAMESE NETWORK DeepFace



$$d(x^{(1)}, x^{(2)}) = \|f(x^{(1)}) - f(x^{(2)})\|_2^2$$

## TRIPLET LOSS FaceNet



$$\text{WANT } \|f(A) - f(P)\|^2 \leq \|f(A) - f(N)\|^2 \Rightarrow d(A, P) - d(A, N) \leq 0$$

BUT WE WANT A GOOD MARGIN, SO...  
 $d(A, P) - d(A, N) + \alpha \leq 0$

HOW DO WE CHOOSE TRIPLETS  
TO TRAIN ON?

- IF A/P ARE VERY SIMILAR, & A/N ARE VERY DIFFERENT  
TRAINING IS VERY EASY.

SELECT A/N THAT ARE PRETTY SIMILAR TO TRAIN A GOOD NET

LEARN THE PARAMS OF  
THE NN SUCH THAT

- IF  $x^i, x^{i'}$  ARE THE SAME  
PERSON  $\cdot d(x^i, x^{i'}) \Rightarrow$  SMALL
- IF  $x^i, x^{i'}$  ARE DIFFERENT  
PEOPLE  $\cdot d(x^i, x^{i'}) \Rightarrow$  LARGE

WE CAN ACCOMPLISH  
THIS WITH THE TRIPLET  
LOSS FUNCTION

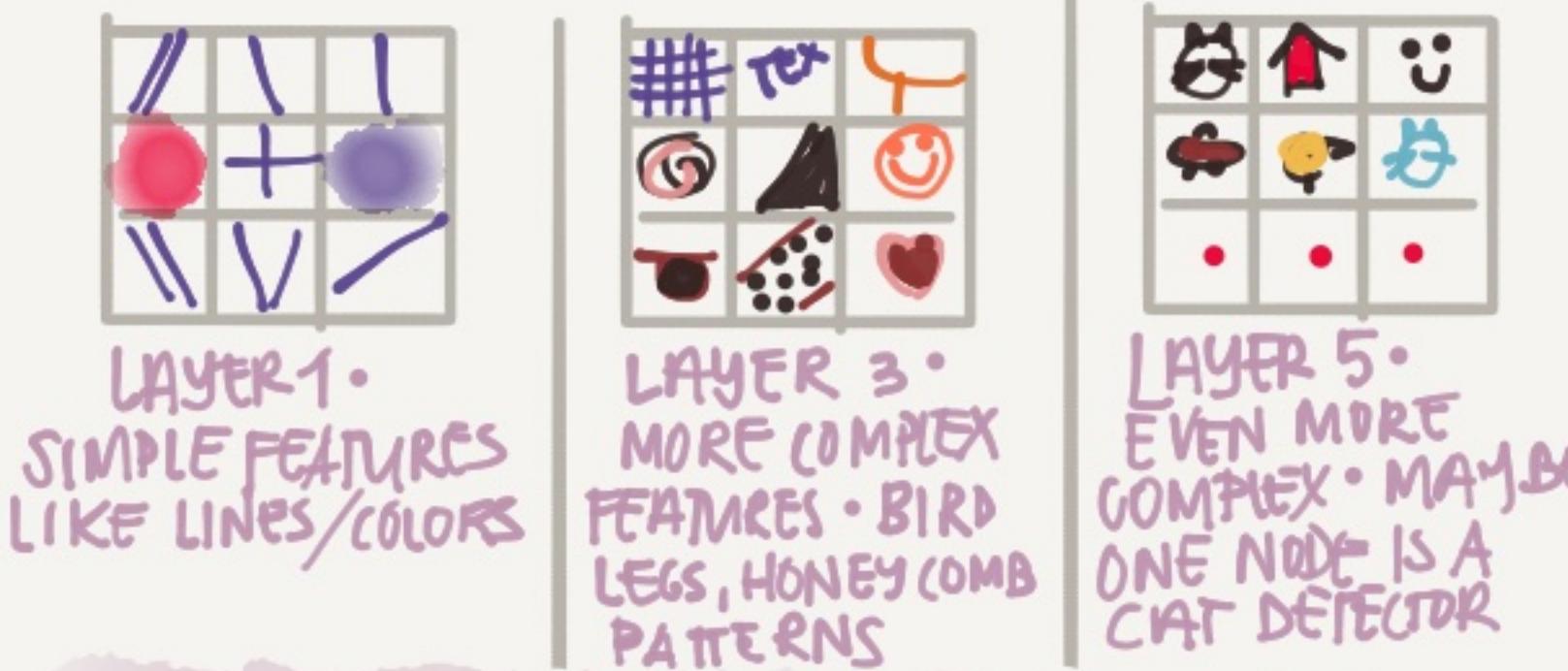
**TIP**: PRECOMPUTE ENCODINGS  
FOR PPL IN YOUR DB, SO YOU  
DON'T HAVE TO SAVE IMAGES  
& COMPUTE ENCODINGS AT RUN-  
TIME

SOME BIG COMPANIES  
HAVE ALREADY TRAINED  
NETWORKS ON LARGE  
AMTS OF PHOTOS SO  
YOU MAY JUST  
WANT TO REUSE  
THEIR WEIGHTS

# NEURAL STYLE TRANSFER



WE CAN VISUALIZE WHAT A NETWORK LEARNS BY LOOKING AT WHAT IMAGES (PARTS) ACTIVATED EACH UNIT MOST



BUT HOW DOES THIS HELP US GENERATE AN IMAGE IN THE STYLE OF ANOTHER?

IDEA:

1. GENERATE A RANDOM IM<sub>G</sub>
2. OPTIMIZE THE COST FUNCTION

$$J(G) = \alpha J_{\text{CONTENT}}(C, G) + \beta J_{\text{STYLE}}(S, G)$$

HOW SIMILAR ARE  $C \text{ & } G$       HOW SIMILAR ARE  $S \text{ & } G$

3. UPDATE EACH PIXEL

## CONTENT COST FUNCTION

- USE A PRE-TRAINED CONVNET (ex VGG)
- SELECT A HIDDEN LAYER SOMEWHERE IN THE MIDDLE  
*LATER  $\Rightarrow$  COPIES LARGER FEATURES*
- LET  $a^{[l]}_i$  &  $a^{[l]}_j$  BE THE ACTIVATIONS
- IF  $a^{[l]}_i$  &  $a^{[l]}_j$  ARE SIMILAR THEY HAVE SIMILAR CONTENT  
*BECAUSE THEY BOTH TRIGGER THE SAME HIDDEN UNITS*

HOW DO WE TELL IF THEY ARE SIMILAR?

$$J_{\text{CONTENT}}(C, G) = \frac{1}{2} \| a^{[l]}_i - a^{[l]}_j \|_F^2$$

## CAPTURING THE STYLE



USING THE STYLE IM<sub>G</sub> AND THE ACTIVATIONS IN A LAYER. LOOK THROUGH THE ACTIVATIONS IN THE DIFFERENT CHANNELS TO SEE HOW CORRELATED THEY ARE

WHEN WE SEE PATTERNS LIKE THIS DO WE USUALLY SEE IT WITH PATCHES LIKE THESE?



## STYLE MATRIX

CREATE A MATRIX OF HOW CORRELATED THE ACTIVATIONS ARE, FOR EACH POS (x,y)  
& CHANNEL PAIR (k, k') FOR THE STYLE IM<sub>G</sub> & GENERATED

$$G_{kk'} = \sum_{i=1}^{n_h} \sum_{j=1}^{n_w} a_{ijk} \cdot a_{ijk'}$$

## THE STYLE COST FUNCTION

$$J(S, G) = \| G^{(S)} - G^{(G)} \|_F^2$$

FROBENIUS NORM

TO GET MORE VISUALLY PLEASING IMAGES IF YOU CALC  $J(S, G)$  OVER MULTIPLE LAYERS



# RECURRENT NEURAL NETWORKS

## SEQUENCE PROBLEMS

IN	OUT	PURPOSE
Mr. Brown	THE QUICK BROWN FOX JUMPED...	SPEECH RECOGNITION
∅	♪ ♪ ♪ ♪ ♪	MUSIC GENERATION
THERE IS NOTHING TO LIKE IN THIS MOVIE	★ ★ ★ ★	SENTIMENT CLASSIFICATION
AGCCCCCTGTG AGGAACCTAG	AGCCCCCTGTG AGGAACCTAG	DNA SEQUENCE ANALYSIS
Voulez-vous chanter avec moi?	Do you want to sing with me?	MACHINE TRANSLATION
🏃‍♂️ 🏃‍♀️ 🏃	RUNNING	VIDEO ACTIVITY RECOGNITION
Yesterday Harry Potter met Hermione Granger	Yesterday Harry Potter met Hermione Granger	NAME ENTITY RECOGNITION

## NAME ENTITY RECOGNITION

$x = \text{HARRY POTTER AND HERMIONE}$   $T_x = 9$   
 $x^{<1>} x^{<2>} \dots$  (9 words)

GRANGER INVENTED A NEW SPELL

$$y = \begin{matrix} 1 & 1 & 0 & 1 & T_y = T_x \\ y^{<1>} & y^{<2>} & \dots & y^{<T>} & \end{matrix}$$

EXAMPLE OF A PROBLEM WHERE  
EVERY  $x^{<i>}$  HAS AN OUTPUT  $y^{<i>}$

## HOW DO WE REPRESENT WORDS?

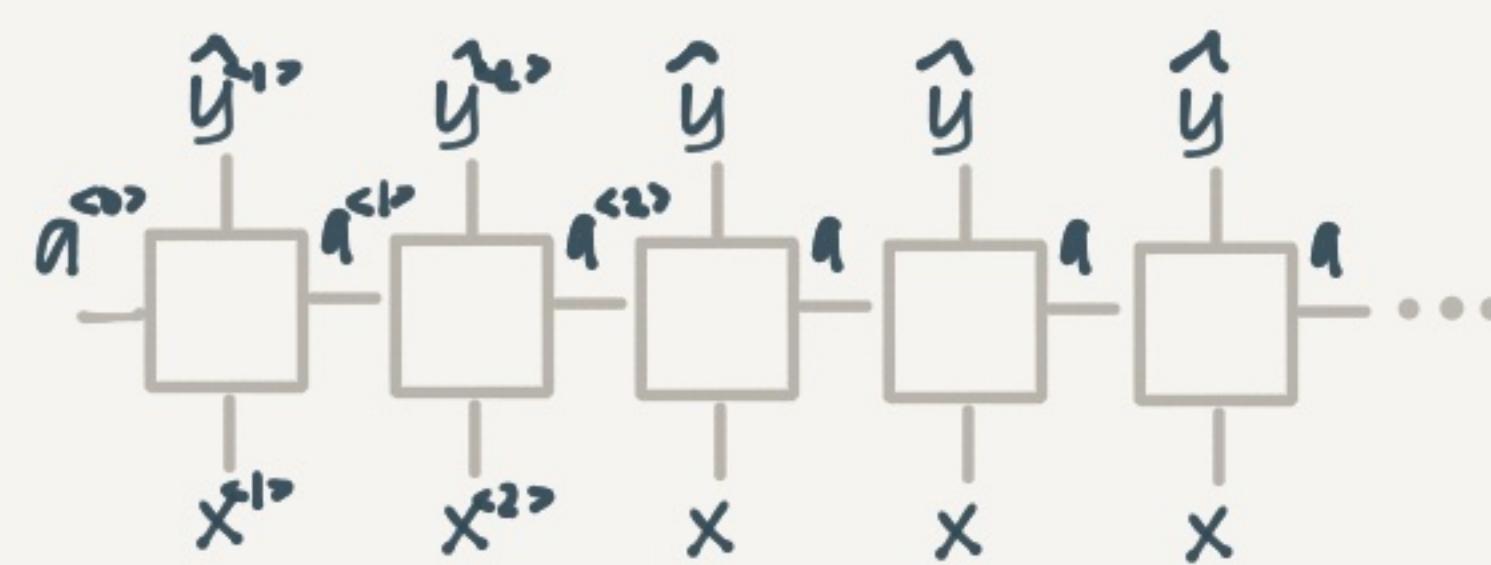
CREATE A VOCABULARY (EG 10K MOST COMMON WORDS IN YOUR TEXTS • OR DOWNLOAD EXISTING)

aaron	1	EACH WORD IS A ONE-HOT.
and	2	VECTOR
Harry	367	$\underline{\text{HARRY}} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$
Potter	4075	
zulu	6830	
	10000	

WE COULD USE A STANDARD NETWORK BUT...

- (A) INPUT & OUTPUTS CAN HAVE DIFFERENT LENGTHS IN DIFF EXAMPLES
- (B) WE DON'T SHARE FEATURES LEARNED ACROSS DIFFERENT POSITIONS

## RECURRENT NEURAL NET (RNN)



PREVIOUS RESULTS ARE PASSED IN AS INPUTS SO WE GET CONTEXT.

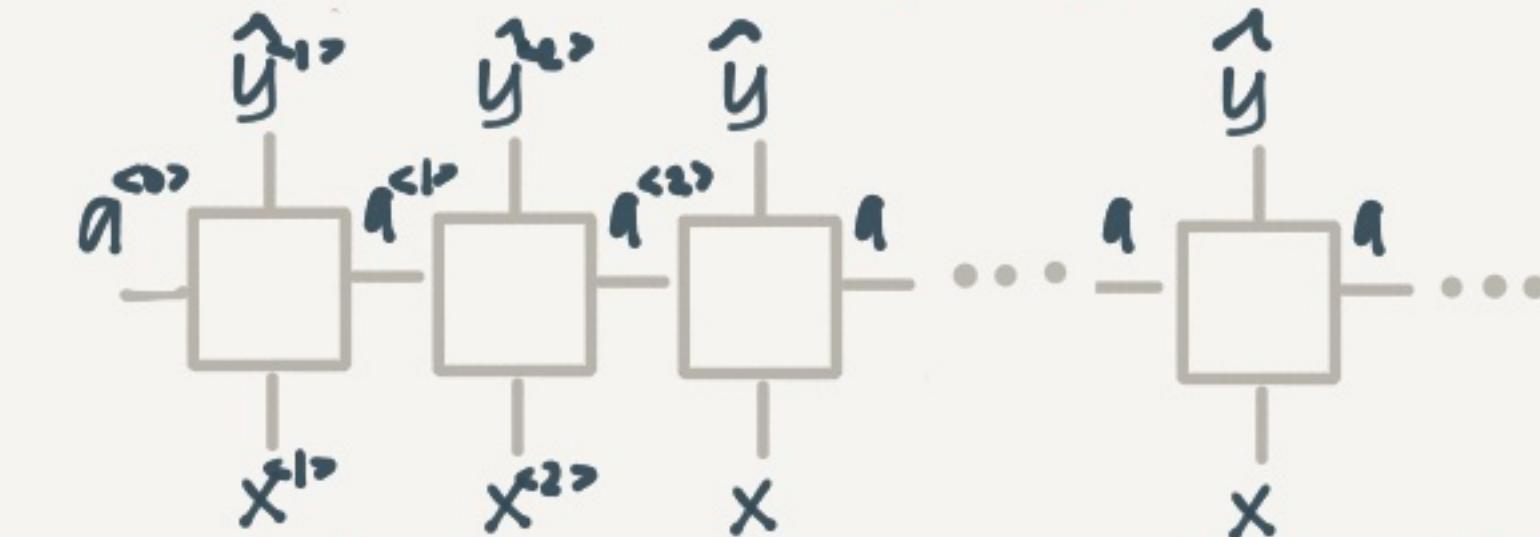
$$\begin{aligned} q^{<1>} &= g_1(W_1[a^{<0>}; x^{<1>}] + b_1) && \text{TANH / RELU} \\ \hat{y}^{<1>} &= g_2(W_{21} q^{<1>} + b_2) && \text{SIGMOID} \end{aligned}$$

THE SAME  $W$  &  $b$  ARE USED IN ALL TIME STEPS

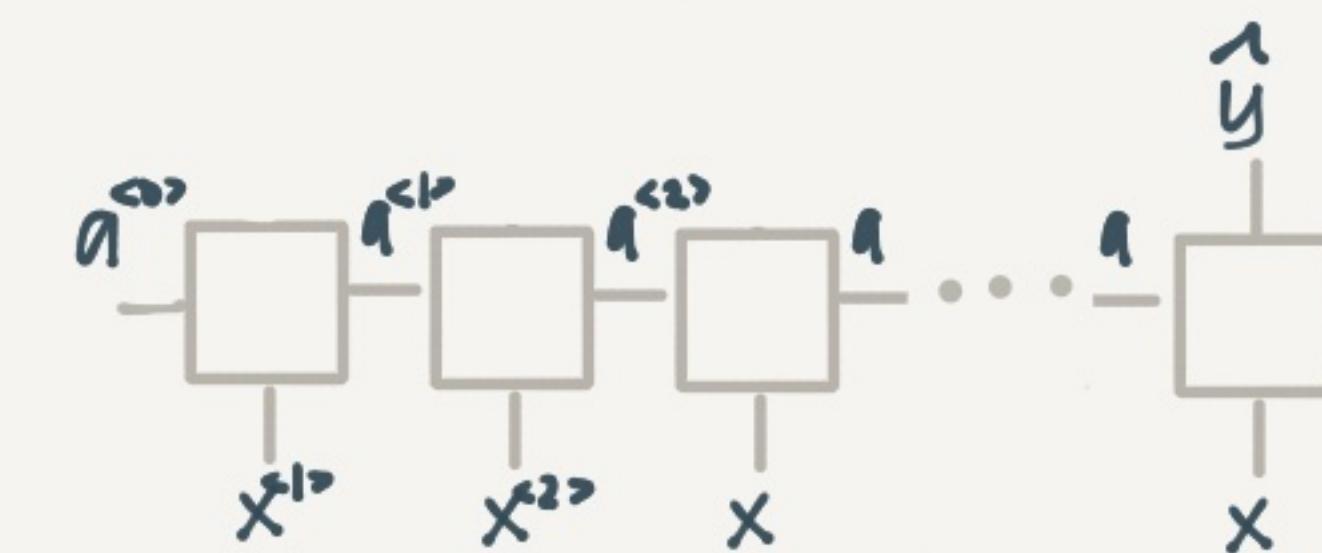
THE LOSS WE OPTIMIZE IS THE SUM OF  $\mathcal{L}(\hat{y}, y)$  FROM 1-T

## DIFFERENT TYPES OF RNN

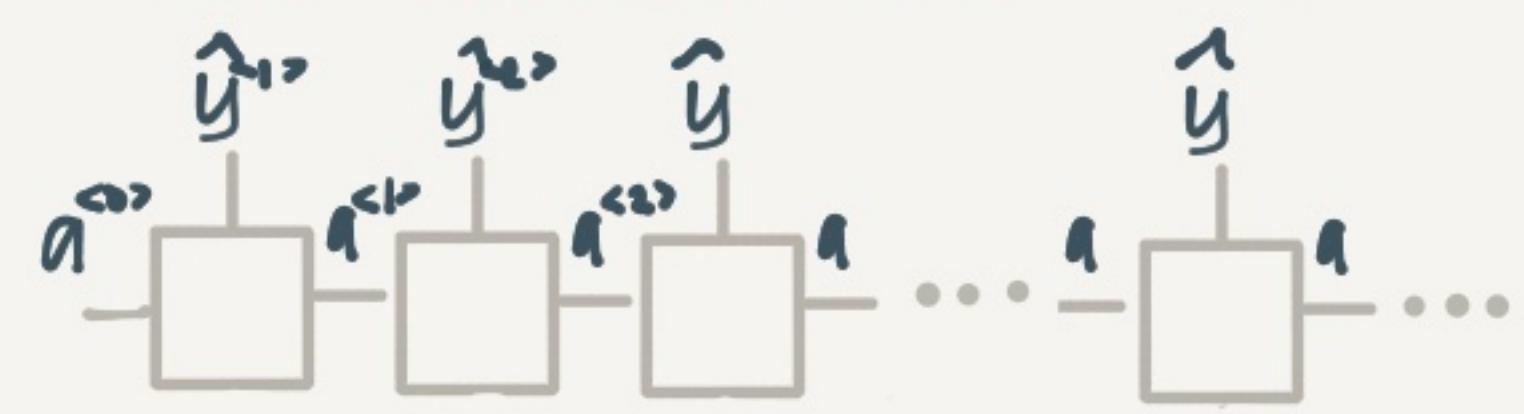
MANY-TO-MANY  $T_x = T_y$



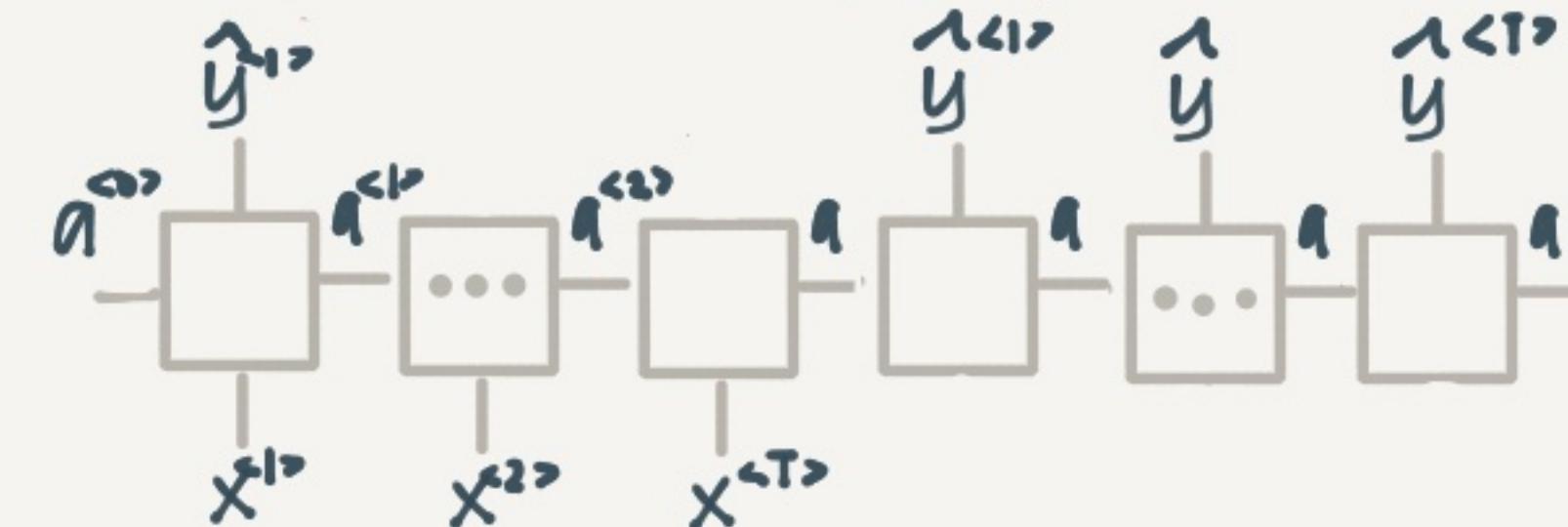
MANY-TO-ONE EX. SENTIMENT ANALYSIS



ONE-TO-MANY • MUSIC GENERATION



MANY-TO-MANY  $T_x \neq T_y$  TRANSLATION



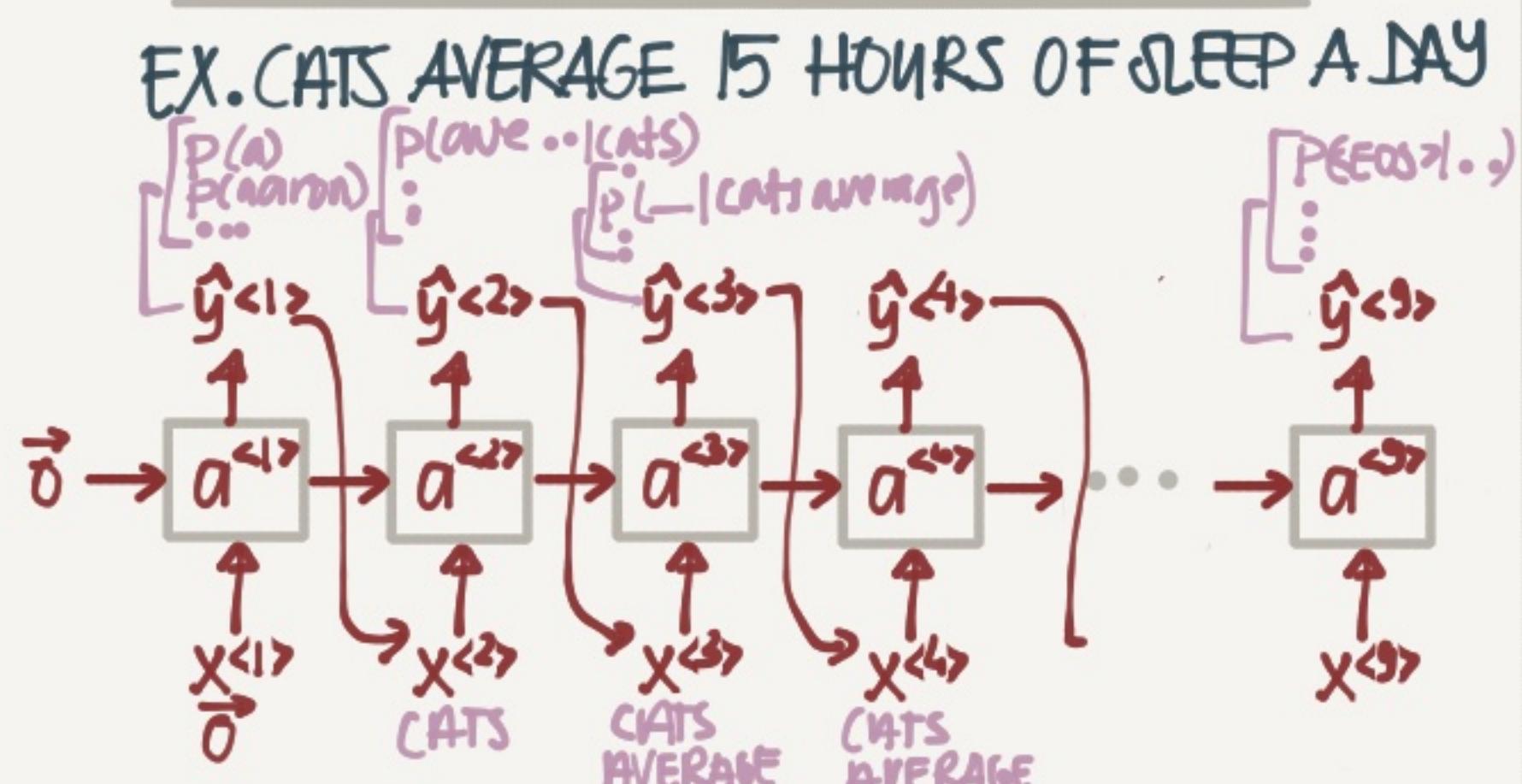
# MORE ON RNNs

## LANGUAGE MODELLING

HOW DO YOU KNOW IF SOMEONE SAID

THE APPLE AND PAIR SALAD OR  
THE APPLE AND PEAR SALAD?

THE PURPOSE OF A LANG. MODEL IS TO  
CALCULATE THE PROBABILITIES



SO GIVEN: CATS AVERAGE IS. WHAT IS THE PROB.  
THE NEXT WORD IS HOURS?

## SAMPLING SENTENCES

1. TRAIN ON ALL HARRY POTTER BOOKS.
2. RANDOMLY SELECT A WORD (ON OF THE TOP WORDS)  
*(EX. THE)*
3. PASS THIS INTO THE NEXT TIMESTAMP  
AND SAMPLE A NEW WORD
4. REPEAT UNTIL X WORDS OR YOU  
REACHED <EOS>

CAN DO AT  
CHARACTER LEVEL  
AS WELL

## VANISHING GRADIENTS

THE CAT, WHO ALREADY ATE APPLES AND ORANGES  
AND A FEW MORE THINGS BUT ~~BU~~ WAS FULL  
THE CATS ~~W~~ ALREADY ATE ...  
... ~~W~~ WERE FULL

NEED TO REMEMBER  
SING/PLURAL FOR A LONG  
TIME

SINCE LONG SENTENCE  $\Rightarrow$  DEEP RNN  
WE GET THE VANISHING GRADIENTS PROB WE  
HAVE IN STANDARD NNs - I.E. THE GRADIENTS  
FOR CAT/CATS HAVE LITTLE OR NO EFFECT  
ON WAS/WERE.

**NOTE:** SOMETIMES YOU SEE EXPLODING GRAD  
(AS OVERFLOW NAN) BUT THIS IS EASILY FIXED  
WITH GRADIENT CLIPPING

GATED RECURRENT UNIT GRU  
HELPS RECALL IF CAT WAS SING.  
OR PLURAL



THE GRU ACTS AS A MEMORY  
— AT EVERY Timestep IT  
CALCULATES A NEW  $\tilde{c}$  TO STORE  
AND A GATE  $\Gamma_u$  DECIDES TO  
UPDATE  $c$  TO  $\tilde{c}$  OR NOT

YAY! YOU ARE NOW  
YOUR OWN J.K. ROWLING

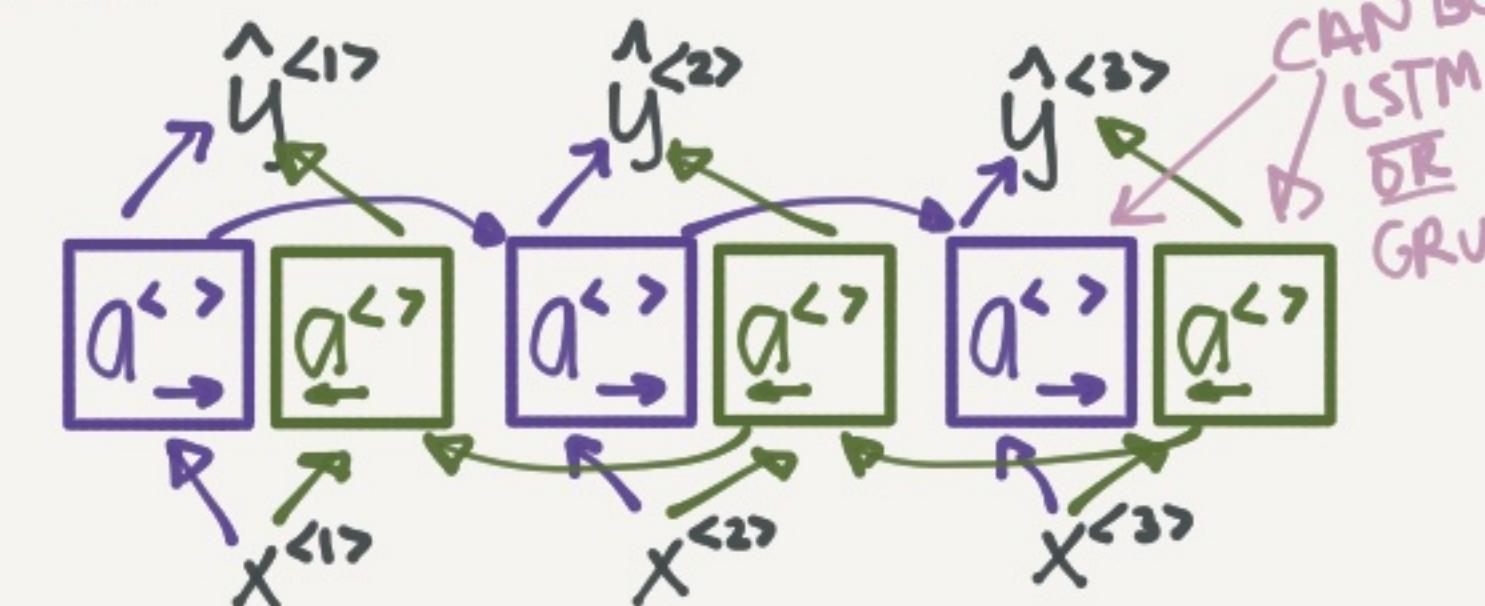
## LONG SHORT TERM MEMORY (LSTM)

THE LSTM IS A VARIATION ON  
THE SAME THEME AS GRU  
BUT WITH AN ADDITIONAL  $\Gamma_f$   
**FORGET GATE**

## BI-DIRECTIONAL RNNs (BRNN)

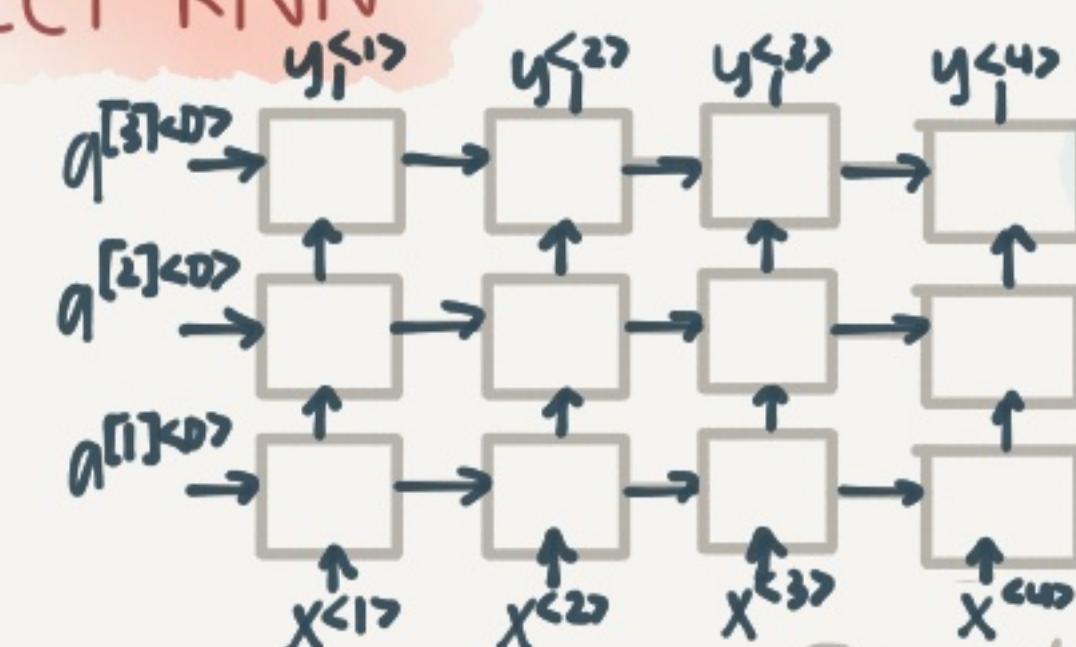
HE SAID, 'TEDDY BEARS ARE ON SALE'  
HE SAID, 'TEDDY ROOSEVELT WAS A  
GREAT PRESIDENT'

PROBLEM: WITHOUT LOOKING FORWARD WE  
CAN'T SAY IF TEDDY IS A TOY OR A NAME



ONE DISADVANTAGE IS THAT YOU NEED  
THE FULL SENTENCE BEFORE YOU BEGIN-  
SO NOT SUITABLE FOR LIVE SPEECH RECO

## DEEP RNN



SINCE THEY  
ARE ALREADY  
MEMORY  
DEEP THEY  
USUALLY  
DON'T HAVE  
A LOT OF  
LAYERS

# NLP & WORD EMBEDDINGS

MAN IS TO WOMAN AS  
KING IS TO QUEEN

PROBLEM: THE ONE-HOT REPR  $\mathbf{q}_\text{apple}$  OF  
APPLE HAS NO INFO ABOUT ITS RELATIONSHIP  
TO  $\mathbf{o}_{\text{orange}}$

I WANT A GLASS OF ORANGE —  
I WANT A GLASS OF APPLE —

SOLUTION: CREATE A MATRIX OF  
FEATURES TO DESCRIBE THE WORDS

## WORD EMBEDDINGS

	MAN	WOMAN	KING	QUEEN	APPLE	ORANGE
5391	1	-0.95	0.97	0.00	0.01	6257
GENDER	-1	1	-0.95	0.97	0.00	0.01
ROYAL	0.01	0.02	0.93	0.95	-0.01	0.00
AGE	0.03	0.02	0.7	0.69	0.03	-0.02
FOOD	0.04	0.01	0.02	0.01	0.95	0.97
:						
$e_{5391}$						

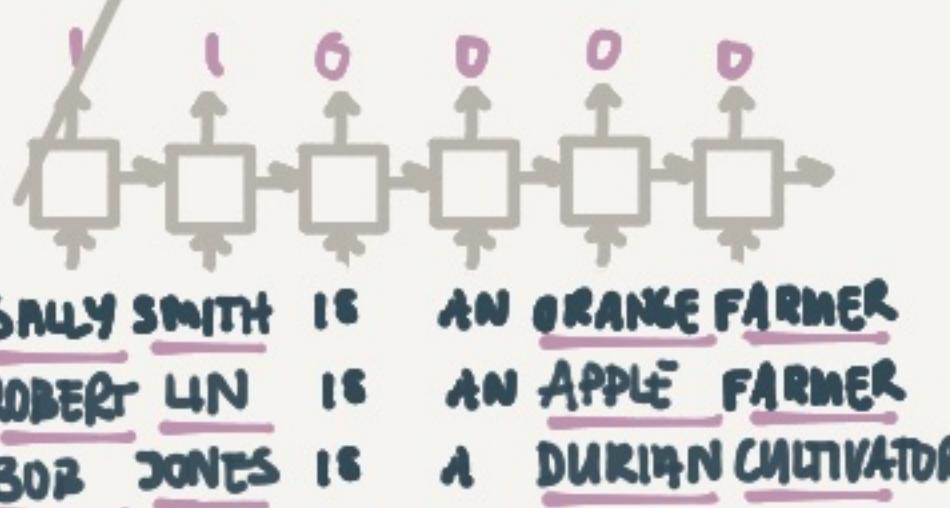
IN REALITY • THE FEATURES ARE  
LEARNED & NOT AS STRAIGHTFWD  
AS GENDER/AGE

• MAN	• woman	• dog
• king	• cat	• fish
• queen	• apple	• orange
• four	• grape	
• three	• orange	
• one		
• two		

t-SNE  
VISUAL  
REPRESENT  
OF 300D  
WORD  
EMBEDDINGS

## USING WORD EMBEDDINGS

EX. NAME/ENTITY RECOGN



WITH WORD EMBEDDINGS WE  
UNDERSTAND THAT AN ORANGE  
FARMER IS A PERSON  $\Rightarrow$  SALLY  
SMITH = NAME

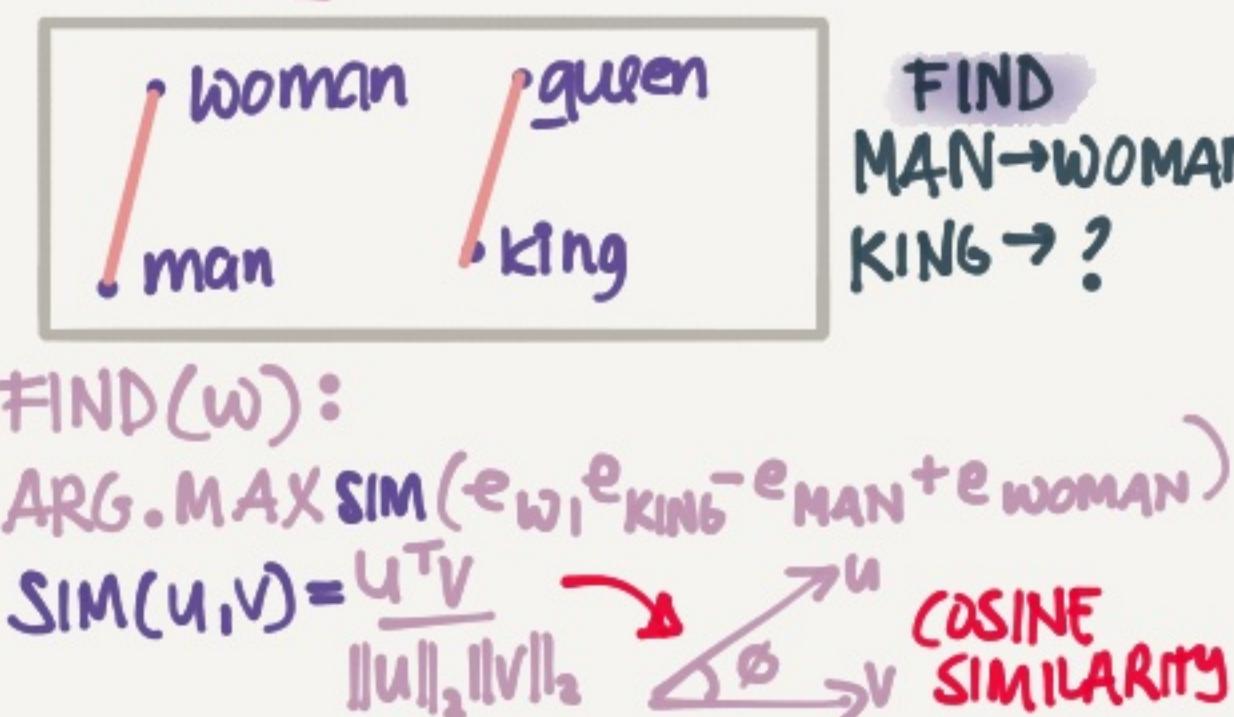
- APPLE ~ ORANGE  $\Rightarrow$  PERSON
- USING WORD EMBEDDINGS TRAINED  
ON LOTS OF TEXT WE ALSO GET EMB.  
FOR MORE UNCOMMON WORDS  
(DURIAN, CULTIVATOR)

EX. MAN IS TO WOMAN AS  
KING IS TO ?

$E = \text{EMBEDDING MATRIX}$

	MAN	WOMAN	KING	QUEEN
5391	1	-0.95	0.97	0.00
GENDER	-1	1	-0.95	0.97
ROYAL	0.01	0.02	0.93	0.95
AGE	0.03	0.02	0.7	0.69
FOOD	0.04	0.01	0.02	0.01
...				
$e_{5391}$				

$$\begin{bmatrix} e_{\text{man}} - e_{\text{woman}} \\ e_{\text{king}} - e_{\text{queen}} \end{bmatrix} \xrightarrow{\text{EVERY SIMILAR}} \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$



## LEARNING WORD EMBEDDINGS

HOW DO WE LEARN THE EMBEDDING MATRIX  $E$ ?

IDEA1: USING A NEURAL LANG MODEL

I WANT A GLASS OF ORANGE  $\hat{y}$



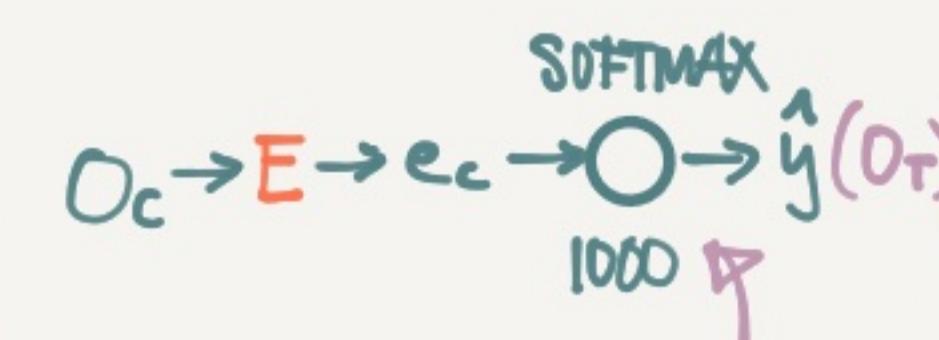
WE CAN USE DIFFERENT CONTEXTS THAN THE LAST 4 WORDS

- LAST 4 WORDS
  - 4 WORDS LEFT+RIGHT
  - LAST 1 WORD
  - NEARBY 1 WORD
- SKIPGRAM**  
RANDOM WITHIN EX 5 WORDS

IDEA2: SKIP-GRAM WORD2VEC

I WANT A GLASS OF ORANGE JUICE TO GO ALONG WITH MY CEREAL  
PICK RANDOM CONTEXT/TARGET PAIRS (WITHIN EX 5 WORDS)

CONTEXT	TARGET
ORANGE	JUICE
ORANGE	GLASS
ORANGE	MY
...	...



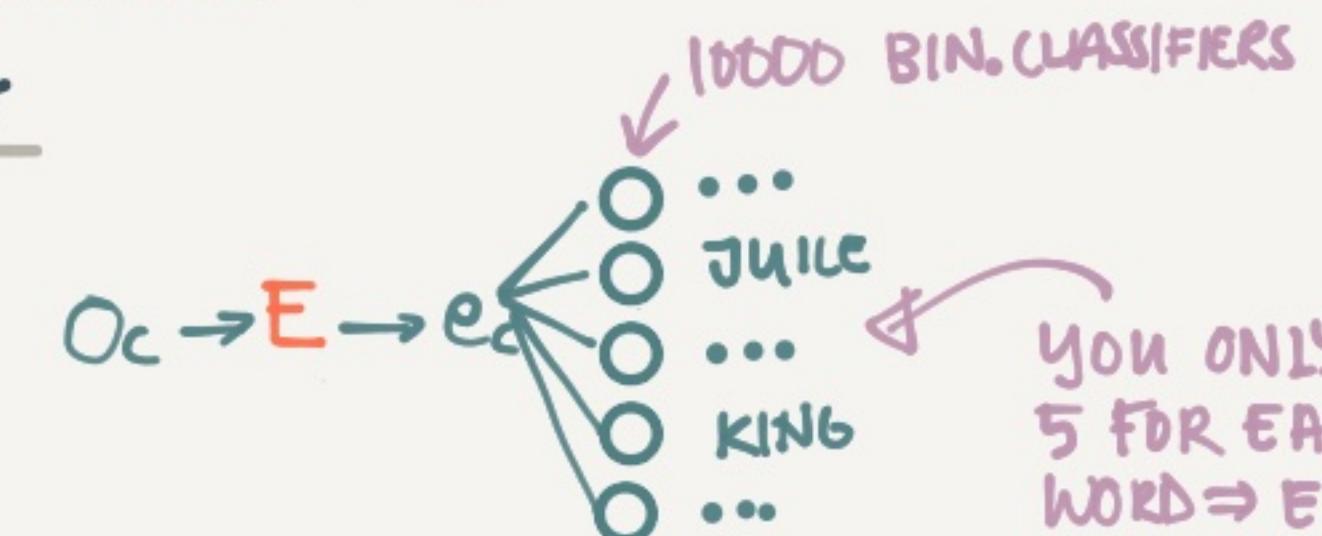
NOTE: WHILE THIS  
SIMPLE NN PREDICTS  $O_t$   
OUR REAL GOAL IS TO  
LEARN  $E$

THIS IS VERY COMPUTATIONALLY EXPENSIVE  
BUT WE CAN OPTIMIZE BY USING A HIERARCHICAL  
SOFTMAX CLASSIFIER

IDEA: NEGATIVE SAMPLING

1. PICK A CONTEXT/TARGET PAIR AS A POSITIVE EXAMPLE
2. PICK A FEW NEG EXAMPLES CONTEXT + RANDOM

CONTEXT	WORD	TARGET
ORANGE	JUICE	1
ORANGE	KING	0
ORANGE	BOOK	0
ORANGE	THE	0
ORANGE	OF	0



NOTE: SOMETIMES BY  
CHANCE YOU PICK A  
POS PAIR • BUT IT DOESN'T  
MATTER

YOU ONLY TRAIN  
5 FOR EACH CONTEXT  
WORD  $\Rightarrow$  EFFICIENT  
TO TRAIN

# WORD EMBEDDINGS

CONTINUED...

## Glove WORD VECTORS

$x_{ij} = \# \text{TIMES WORD } i \text{ APPEARS IN THE CONTEXT OF } j$

TARGET CONTEXT  
(HOW RELATED THEY ARE)

$$\text{MINIMIZE } \sum_{i=1}^{10k} \sum_{j=1}^{10k} f(x_{ij})(\theta_i^T e_j + b_i + b_j - \log x_{ij})^2$$

IF NO CONTEXT  
(ALSO HELPS WEIGHING VERY FREQ WORDS (THE, OF...) & VERY INFREQUENT (PURPLE))

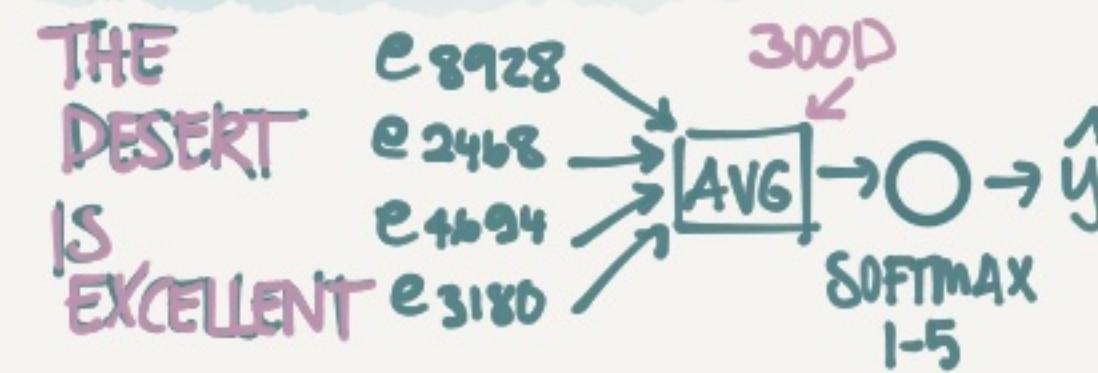
EVERYTHING LED UP TO THIS VERY SIMPLE ALGORITHM

## SENTIMENT CLASSIFICATION

X	Y
THE DESSERT IS EXCELLENT	★★★☆
SERVICE WAS QUITE SLOW	★☆
GOOD FOR A QUICK MEAL BUT NOTHING SPECIAL	★☆☆
COMPLETELY LACKING IN GOOD TASTE, GOOD SERVICE AND GOOD AMBIENCE	*

PROBLEM: YOU MAY NOT HAVE A LARGE DATASET  
BUT YOU CAN USE AN EMBEDDING MATRIX  $E$  THAT IS ALREADY PRE-TRAINED

### IDEA: SIMPLE CLASSIFICATION

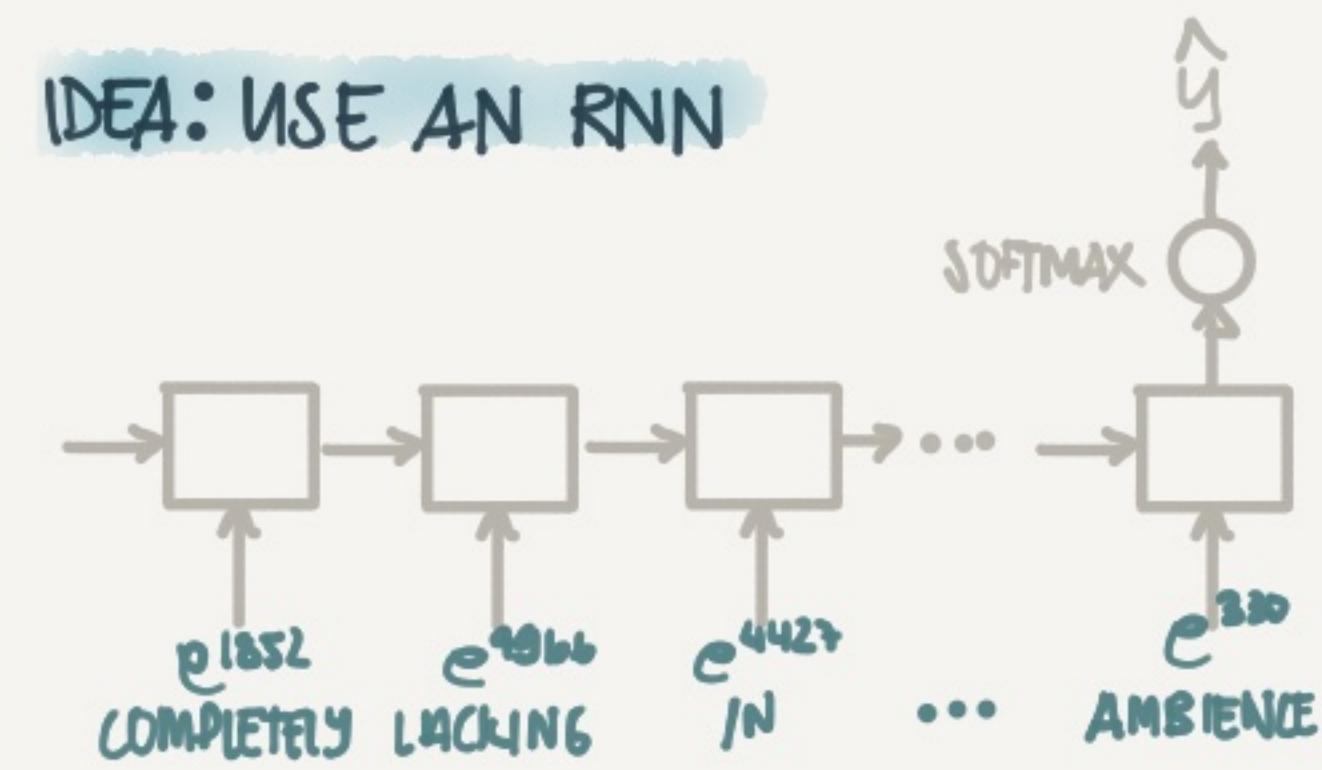


WORKS WELL FOR SHORT SENTENCES  
BUT DOESN'T TAKE ORDER INTO ACCOUNT

"COMPLETELY LACKING IN GOOD TASTE,  
GOOD SERVICE AND GOOD AMBIENCE"

THIS MAY BE SEEN AS A ~~++~~ REVIEW

### IDEA: USE AN RNN



THIS CAN NOW TAKE INTO ACCOUNT THAT COMPLETELY LACKING NEGATES THE WORD GOOD

## ELIMINATING BIAS IN WORD EMBEDDINGS

MAN IS TO COMPUTER PROGRAMMER AS WOMAN IS TO HOME MAKER

SOMETIMES THE TEXT CONTAINS ♂ ALBOS LEARN A GENDER, RACE, AGE... BIAS WE DON'T WANT OUR MODELS TO HAVE. EX. HIRING BASED ON GENDER, SENTENCING BASED ON RACE ETC.

## ADDRESSING BIAS

### 1. IDENTIFY BIAS DIRECTION



### 2. NEUTRALIZE

FOR EVERY WORD THAT IS NOT DEFINITIONAL (GIRL, BOY, HE, SHE...) PROJECT TO GET RID OF BIAS

### 3. EQUALIZE PAIRS

THE ONLY DIFF BETWEEN EX GIRL/BOY SHOULD BE GENDER

HOW DO YOU KNOW WHICH WORDS TO NEUTRALIZE?

DOCTOR, BEARD, SEWING MACHINE?

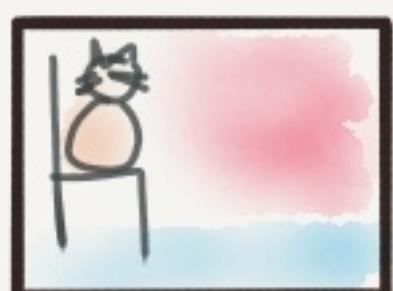
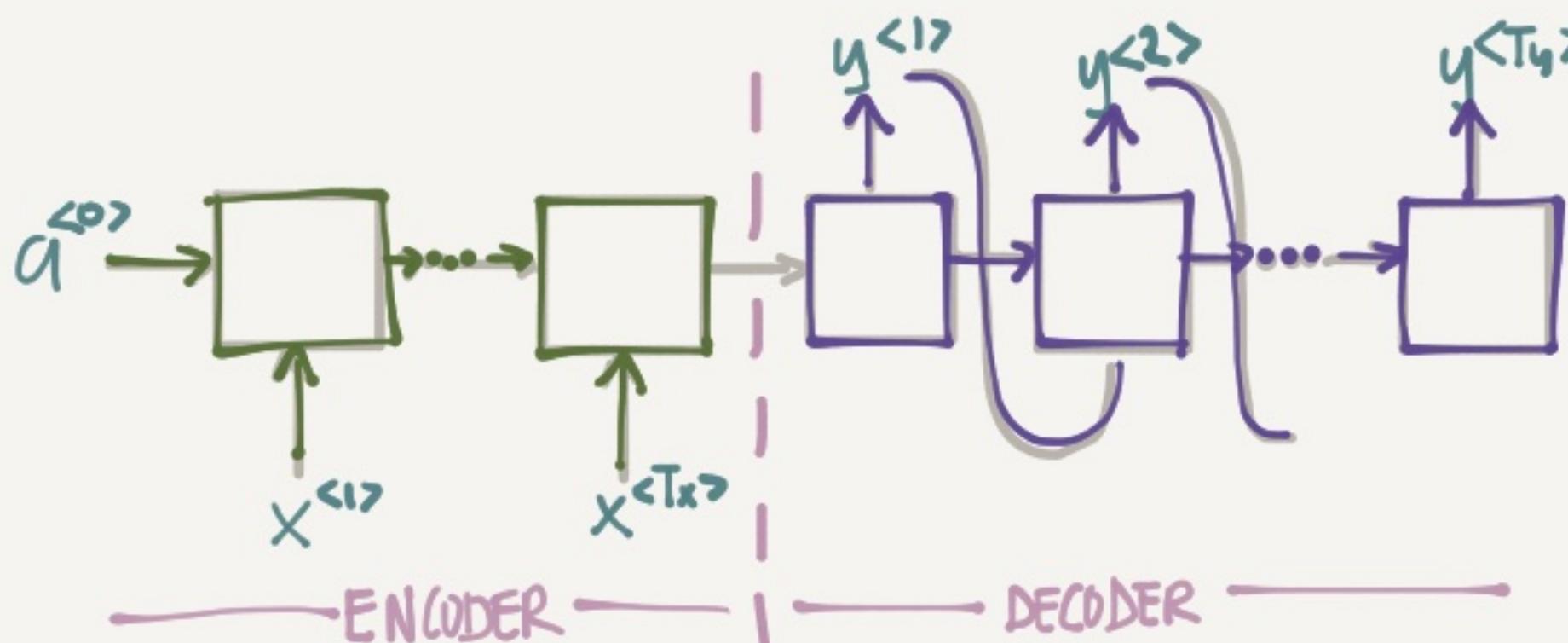
A: BY TRAINING A CLASSIFIER TO FIND OUT IF A WORD IS DEFINITIONAL

TURNED OUT THE # OF PAIRS IS FAIRLY SMALL SO YOU CAN EVEN HAND PICK THEM

# SEQUENCE TO SEQUENCE

## BASIC MODELS

JANE VISITE L'AFRICA  
EN SEPTEMBRE → JANE IS VISITING AFRICA  
IN SEPTEMBER



→ THIS IS A CAT  
ON A CHAIR

CNN → RNN

HOW DO YOU PICK THE MOST LIKELY SENTENCE?

$$P(y^{<1>}, \dots, y^{<T_y>} | x)$$

WE DON'T WANT A RANDOMLY GENERATED SENTENCE  
(WE WOULD SOMETIMES GET A GOOD, SOMETIMES BAD)  
INSTEAD WE WANT TO MAXIMIZE

$$\text{ARG MAX } P(y^{<1>}, \dots, y^{<T_y>} | x)$$

IDEA: USE GREEDY SEARCH

1. PICK THE WORD WITH THE BEST PROBABILITY
2. REPEAT UNTIL DEAD

WITH THIS WE COULD GET

- JANE IS GOING TO BE VISITING AFRICA  
THIS SEPTEMBER

INSTEAD OF

- JANE IS VISITING AFRICA THIS SEPTEMBER

SOLUTION

OPTIMIZE THE PROB OF THE WHOLE SENTENCE INSTEAD

## BEAM SEARCH

1. PICK THE FIRST WORD

PICK THE B (EX 3) BEST ALTERNATIVES  
(IN, JANE, SEPTEMBER)

2. FOR EACH B WORDS PICK THE NEXT WORD AND EVALUATE THE PAIRS TO END UP w B PAIRS

$$P(y^{<1>} | x) = P(y^{<1>} | x) P(y^{<2>} | x, y^{<1>})$$



(IN SEPTEMBER, JANE IS, JANE VISITS)

3. REPEAT TIL DONE

$$\text{ARG MAX } \prod_{t=1}^{T_y} P(y^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})$$

OVERFLOWS

PROBLEM: MULTIPLYING PROBABILITIES ( $0 < p \ll 1$ )  
RESULTS IN A VERY SMALL NUMBER

PROBLEM II: IF WE OPTIMIZE FOR THE MULT  
WE WILL PREFER SHORT SENTENCES. SINCE  
EACH WORD WILL REDUCE PROB

INSTEAD WE CAN OPTIMIZE FOR THIS

$$\frac{1}{T_y} \alpha \sum_{t=1}^{T_y} \log(P^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})$$

HOW DO WE PICK B?

LARGE B: BETTER RESULT, SLOWER  
SMALL B: WORSE RESULT, BETTER

IN PROD YOU MIGHT SEE B=10.  
100 IS PROBABLY A BIT TOO HIGH -  
BUT ITS DOMAIN DEPENDENT

ERROR ANALYSIS IN BEAM S.

HUMAN: JANE VISITS AFRICA IN SEPT... y\*  
ALSO: JANE VISITED AFRICA LAST SEPTEMBER y

HOW DO WE KNOW IF ITS OUR RNN  
OR OUR BEAM SEARCH WE SHOULD  
WORK ON?

$$\text{LET THE RNN GIVE } P_y^* = P(y^*, x) \text{ & } P_y^{\hat{*}} = P(y^{\hat{*}}, x)$$

IF  $P_y^* > P_y^{\hat{*}}$ :

BEAM PICKED THE WRONG ONE  
TRY A HIGHER B

ELSE:

THE RNN PICKED THE WRONG  
PROBS - SO FOCUS ON THE RNN

# SEQUENCE TO SEQUENCE

FRENCH: LE CHAT EST SUR LE TAPIS  
 HUMAN1: THE CAT IS ON THE MAT  
HUMAN2: THERE IS A CAT ON THE MAT

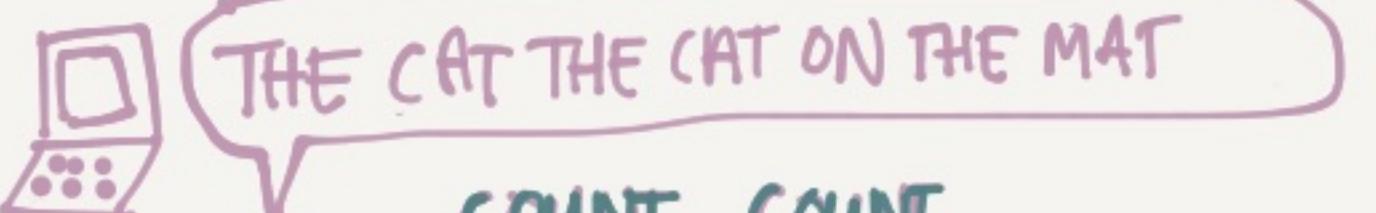
How do you evaluate the machine translation when multiple are right?

## BLEU SCORE

IDEA: CHECK IF THE WORDS <sup>MR</sup> APPEAR IN THE REAL TRANSLATION



IDEA: ONLY GIVE CREDIT FOR A WORD THE MAX # TIMES IT APPEARS IN A TARGET SENTENCE  
 SCORE: 2/7 - COUNT CLIP



BIGRAMS

## COMBINED BLEU SCORE

$$BP \cdot \exp\left(\frac{1}{4} \sum_{n=1}^t p_n\right)$$

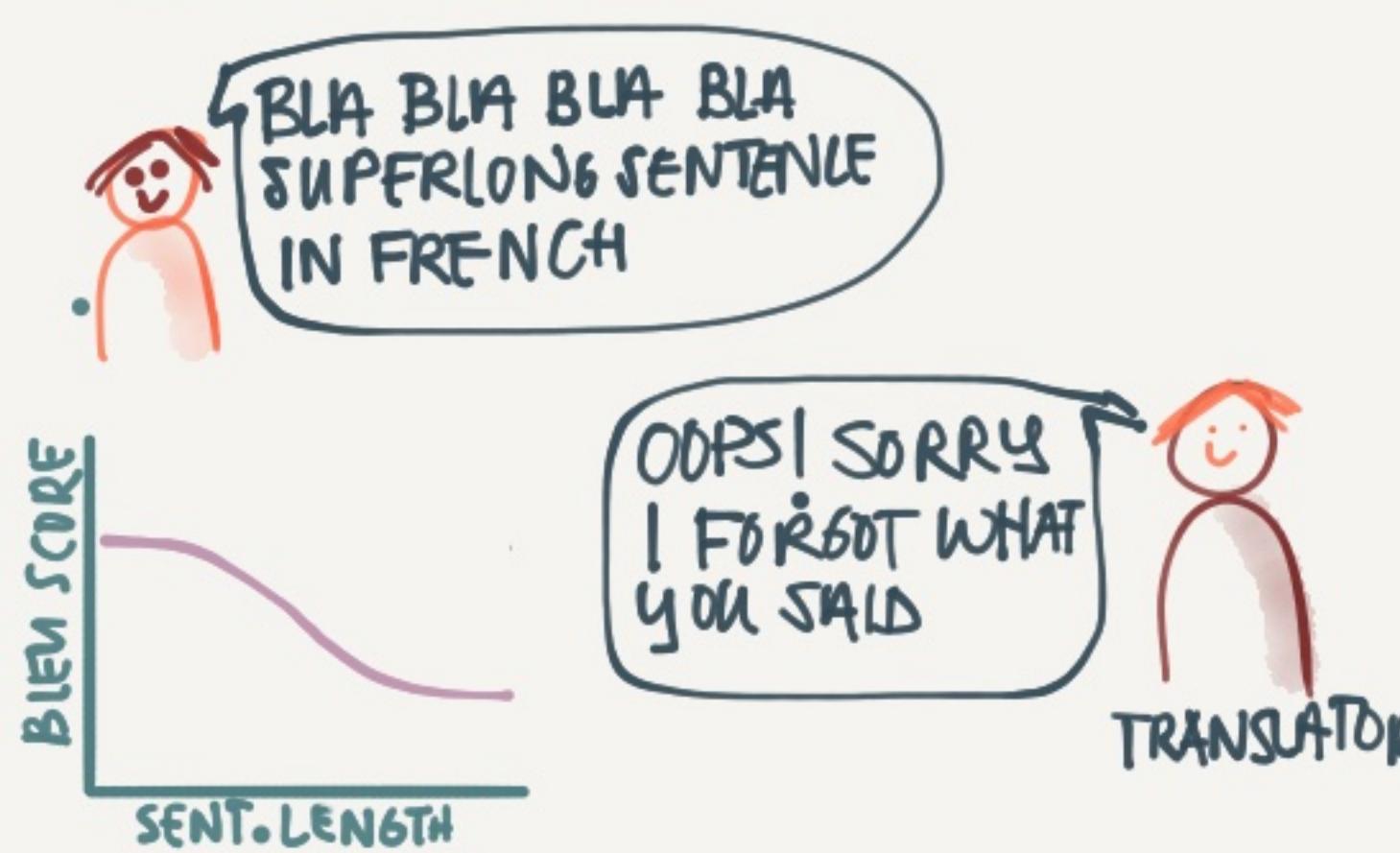
$p_1$  = SCORE SINGLE WORD  
 $p_2$  = SCORE BIGRAMS

...  
 BP = BREVITY PENALTY

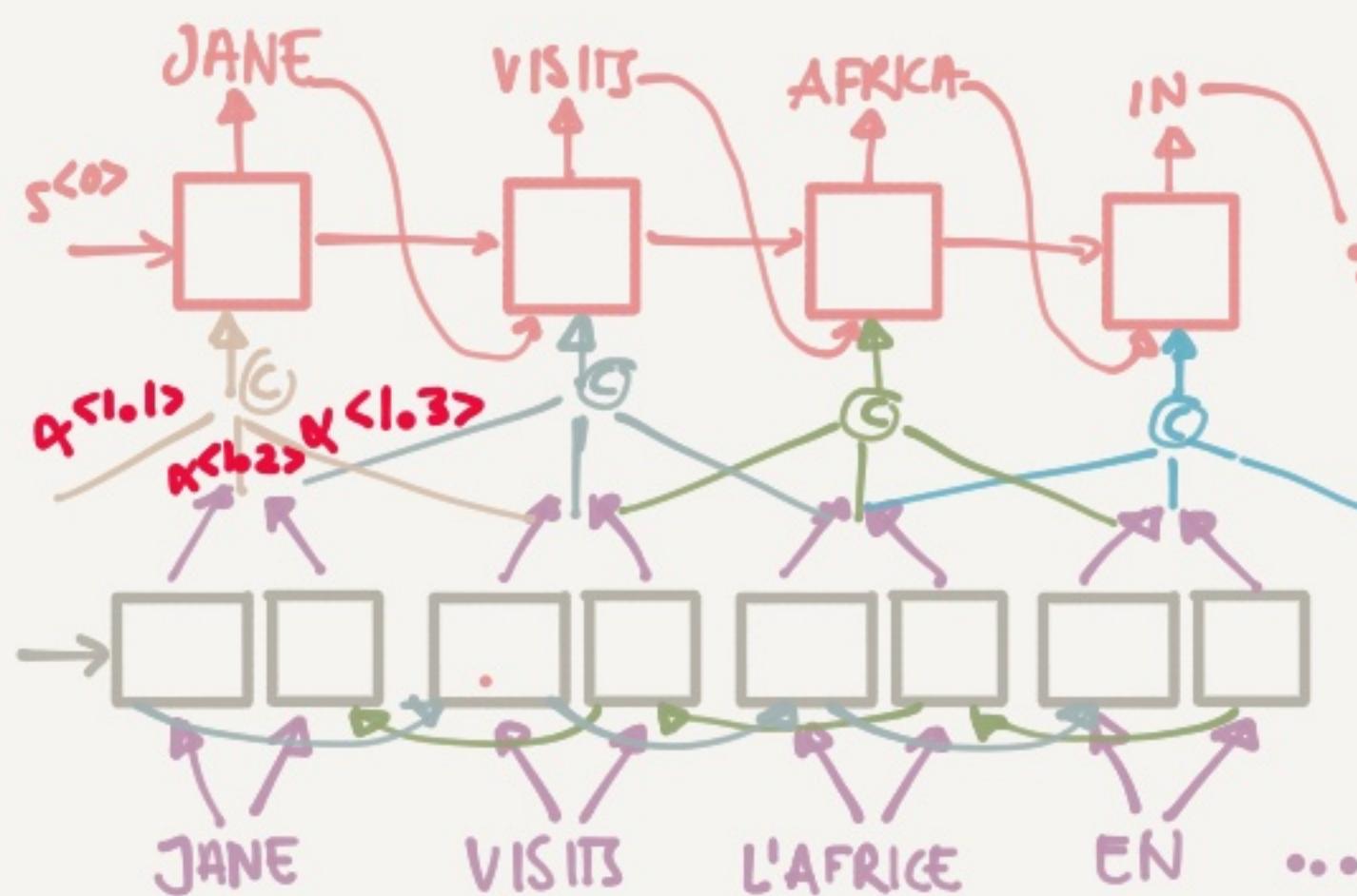
PENALIZES  
SENTENCES  
SHORTER  
THAN THE  
TARGET

A USEFUL SINGLE NUMBER  
EVAL METRIC

## ATTENTION MODEL



SOLUTION: TRANSLATE A LITTLE AT A TIME USING ONLY PARTS OF THE SENTENCE AS CONTEXT



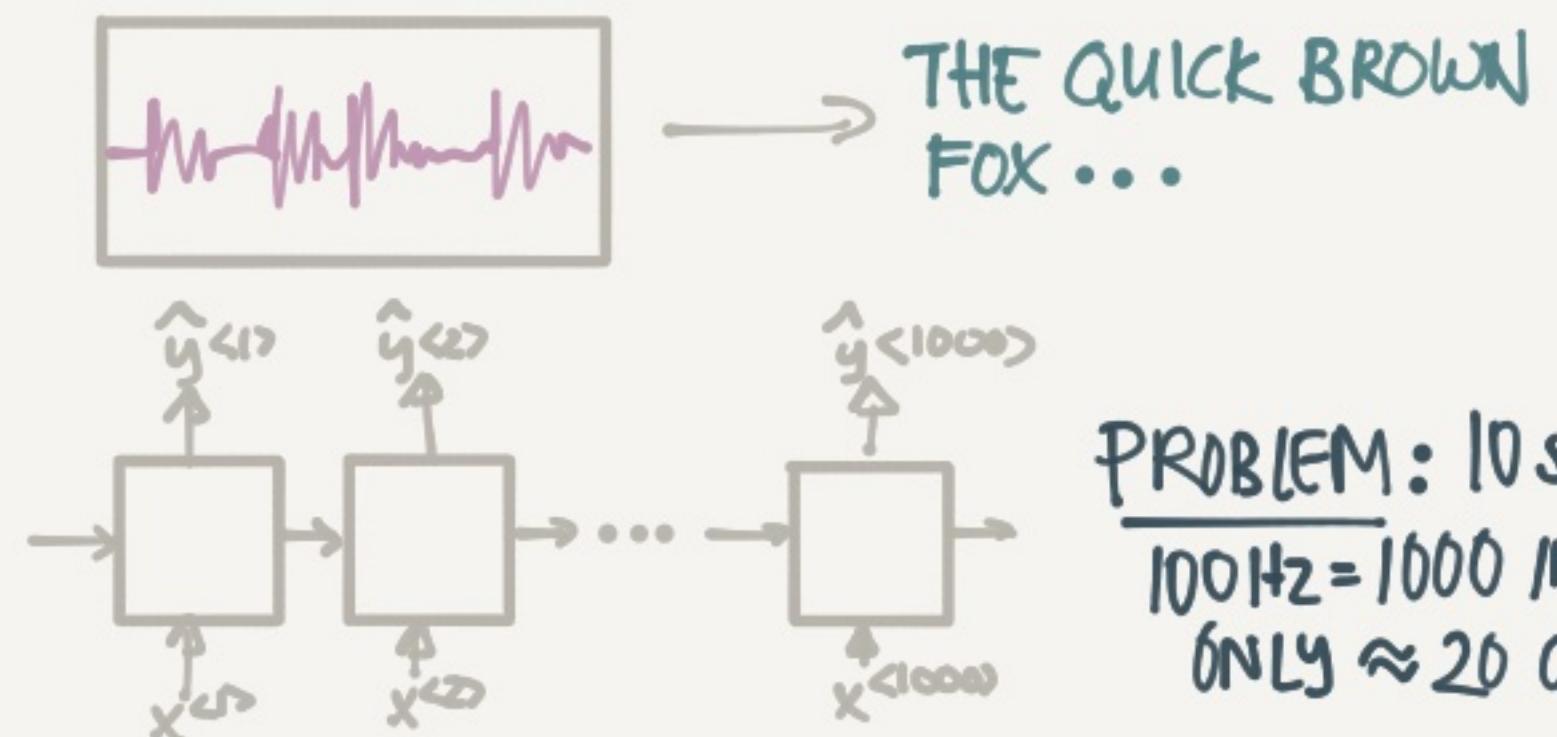
$$\alpha^{<t,t>} = \text{HOW MUCH ATTENTION } y^{<t>} \text{ SHOULD PAY TO } a^{<t>}$$

$$C^{<2>} = \sum_{t'} \alpha^{<2,t>} \cdot a^{<t>} \quad \sum_t \alpha^{<1,t>} = 1$$

$\alpha$  IS CALCULATED USING A SMALL NEURAL NETWORK

$$s^{<t-1>} \rightarrow e^{<t,t>} \quad \alpha^{<t,t>} = \frac{\exp(e^{<t,t>})}{\sum_{t'=1}^T \exp(e^{<t,t>})}$$

## SPEECH RECOGNITION



SOLUTION: USE CTC COST (CONNECTION TEMPORAL CLASSIFICATION)

t t - h \_ e e e - - - l u - - - q q q - - - o

COLLAPSE REPEATED CHARS NOT SEP BY BLANK

## TRIGGER WORD DETECTION

